THEORY AND ALGORITHMS FOR COMMUNITY DETECTION AND CLUSTERING IN NETWORKS

A Dissertation

by

EUGENE LYKHOVYD

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Sergiy Butenko
Committee Members,	Jianer Chen
	Kiavash Kianfar
	Lewis Ntaimo
Head of Department,	Mark Lawley

August 2019

Major Subject: Industrial Engineering

ABSTRACT

This dissertation is focused on certain clustering and partitioning problems in networks. We present a comprehensive study of the maximum independent union of cliques problem and its generalizations in uniform random graphs. The main result is the logarithmic upper bound, similarly to the maximum clique, which suggests a subexponential algorithm. Then we propose a parallel version of Russian Doll Search, an algorithm that can be used to find the maximum independent union of cliques. We enhance existing verification procedure for this problem by a simple observation, which also leads to an elegant constant time biclique verification. Finally, we perform the first computational study for finding Hadwiger's number of a graph. We present several integer formulations, scale-reduction techniques, heuristics, and bounds, together with a scheme for future exact combinatorial algorithms.

DEDICATION

To my parents who always believed in me

ACKNOWLEDGMENTS

I would like to thank everyone who assisted me in writing this dissertation. Special thanks to my parents for igniting the passion for the research in my heart. Without you this would not be possible. I gratefully thank my advisor, Sergiy Butenko, for leading me along this thorny academic path, believing in my success, making me seeing graphs everywhere, and, importantly, granting freedom to advance my research. In addition, I want to thank every member of my committee, as from each of you I took one or more classes, thank you for your superior teaching and valuable comments about this work. Also, shoutout to my two lovely guinea pigs for their moral support. And lastly, I acknowledge Texas A&M University for the wonderful atmosphere, traditions, and uprightness of conduct that each and every Aggie is taught to carry on in his or her life.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professors Sergiy Butenko, the chair and advisor, Kiavash Kianfar, and Lewis Ntaimo of the Department of Industrial and Systems Engineering, and Professor Jianer Chen of the Department of Computer Science and Engineering.

The study depicted in Section 4 was conducted in part by Austin Buchanan of the Department of Industrial Engineering & Management of Oklahoma State University.

All other work conducted for the dissertation was completed by the student independently. Portions of this research were conducted with the advanced computing resources provided by Texas A&M High Performance Research Computing.

Funding Sources

Graduate study was supported by a fellowship from the Department of Industrial and Systems Engineering of Texas A&M University as well as graduate assistantship from Dr. Sergiy Butenko.

This work was made possible in part by NSF under Grant CMMI-1538493 and AFOSR award FA9550-19-1-0161. Its contents are solely the responsibility of the author and do not necessarily represent the official views of the institutions above. The support of AFRL Mathematical Modeling and Optimization Institute is also acknowledged.

NOMENCLATURE

IUC	Independent Union of Cliques
IUΠ	Independent Union of Π-sets
RDS	Russian Doll Search
LP	Linear Program
IP	Integer Program
MIP	Mixed Integer Program
ER	Erdős-Rényi graph
WLOG	Without Loss of Generality

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Many complex real-life systems can be efficiently simulated in terms of Graph Theory. In these systems, the components, subcomponents, or logically separate parts may be viewed as network nodes, when the relations between these components may be described as arcs between respective nodes.

Historically the first network is thought to be introduced by Swiss mathematician Leonhard Euler back in 1735, where he solved the following graph traversing problem. The city of Königsberg had several islands with seven bridges in between (Figure[1] 1.1). Euler had an interesting question: "Could a person visit every island and return home crossing each of the seven bridges only once?" This was an impossible question for that time, as tools and principles of modern mathematics, especially graph theory, were not well developed and understood. In his work [3] Euler built the network with islands as nodes, arcs as bridges, and presented a rigorous proof with the negative answer as well as discovered the famous formula $V - E + F = 2$, where V is the number of nodes in the network, E is the number of arcs, and F is the number of faces.

There are many other examples of such structures. The first naïve example is a *transportation network*, where nodes represent different locations, e.g., warehouses, airports, seaports, certain places of interest, while arcs represent connectivity between two different locations, e.g., shipment routes, airways, roads. The other "traditional" example for analysis is a *social network*, where nodes represent people or groups, and arcs relate to social interaction. One of the interests of analyzing social networks is determining coherent groups or clusters, also called *cliques*, because they might reveal undiscovered information about an individual. An example is author's Facebook friends network, seen[2] in Figure 1.2, where a highly-connected node #1 is the author's girlfriend, other connections are from high school in Ukraine creating cluster #2, undergrad school in Ukraine for cluster #3, coworkers for cluster #4, Ukrainian friends in the USA for cluster #5, and grad

[1] Sketch by Matthäus Merian in 1652; modified by Bogdan Giuşcă and distributed under GFDL.
[2] Generated by `https://lostcircles.com/`.

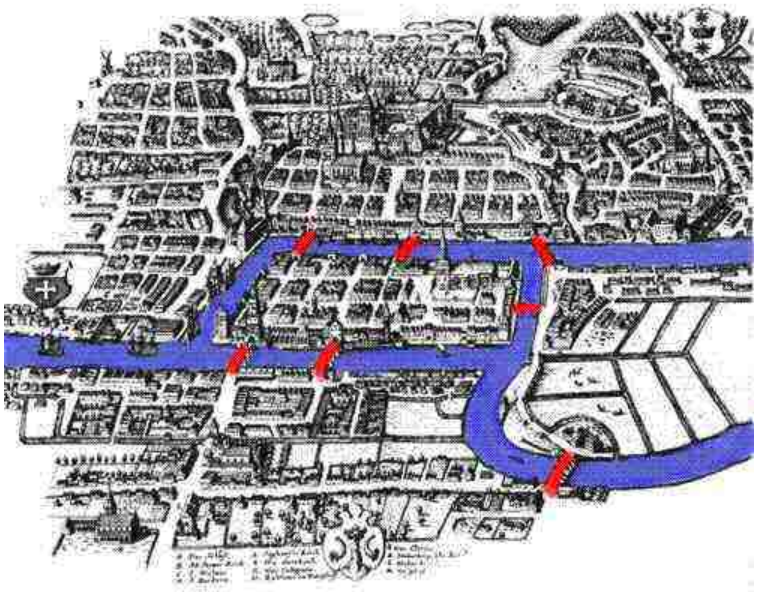


Figure 1.1: Map of Königsberg in 17th century

school for cluster #6. More "exotic" examples are financial networks [4], proteins interactions networks [5], materials networks [6] and many others.

Nowadays businesses generate tones of data and want to be able to find interesting insights about their performances. This causes the demand for jobs with "buzzing" skills such as data mining, big data analysis, deep machine learning, and others; often many such problems may be solved utilizing graph theoretical toolkit. In this dissertation we present theoretical studies as well as algorithms with computational results about graph clustering problems.

1.1 Preliminaries

In Section 1.2 we present the summary of contributions of this dissertation. However, before that, the reader may need to become familiar with terminology and basic concepts discussed thereof. This section introduces necessary terminology, optimization principles, random graphs, and integer programming. More details will be provided to the reader whenever needed.

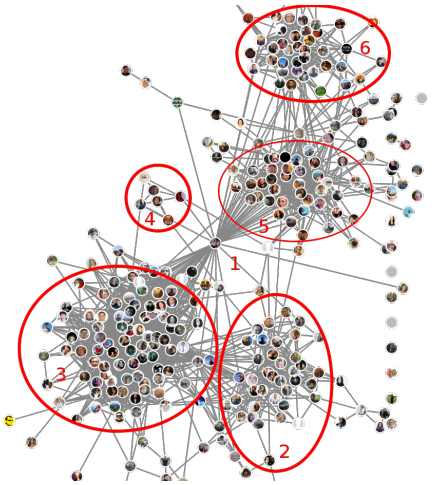


Figure 1.2: Facebook friendship network of the author

1.1.1 Graph theory terminology

The main problems discussed in this dissertation use the concept of a graph. There are two standard vocabularies to talk about such objects in the literature: 1) a network with nodes and arcs; 2) a graph with vertices and edges. We will adhere to the latter one, however they may be used interchangeably in other scientific works.

A *graph* is a pair of a vertex set and an edge set, denoted as $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of n vertices and E is the edge set. If the edge set is a subset of $V \times V$, i.e., Cartesian product or vertex pairs with order, then the graph is called *directed*, or if it is a subset of $\binom{V}{2}$, i.e., all vertex pairs without order, then the graph is called *undirected*. We call a graph to be *simple*, when it contains no loops, i.e., edges with coinciding endpoints, as well as multiedges, i.e., multiple edges between two vertices. In this work all graphs considered are simple and undirected, unless stated otherwise. Later on, the notation $\binom{S}{2}$ for a set S means $\{\{v, w\} \mid v \in S, w \in S, v \neq w\}$.

For the edge $\{v, w\} \in E$, vertices $v \in V$ and $w \in V$ are called *adjacent* or *neighbors* while the edge $\{v, w\}$ is called *incident* to v and w. The set of all neighbors of a vertex $v \in V$ is called the *neighborhood* of v and denoted as $N_G(v) = \{w \mid \{v, w\} \in E\}$, and its cardinality $d_G(v) = |N_G(v)|$ is called the degree of a vertex v. The minimum and maximum degree in G is denoted by

$\delta(G)$ and $\Delta(G)$, respectively. A graph $G' = (V', E')$ is a *subgraph* of graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. Given a subset of vertices $S \subseteq V$, the corresponding *induced subgraph* is the graph obtained from G by deleting all vertices $V \setminus S$ together with edges incident to at least one of them, denoted by $G[S]$. We call two graphs $G = (V, E)$ and $G' = (V', E')$ *isomorphic* if there exists a bijective map $f : V \to V'$ such that $E' = \{\{f(v), f(w)\} \mid \{v, w\} \in E\}$, i.e., they differ only in vertex labeling. We call two graphs *non-isomorphic* otherwise.

A *path* of length p between vertices v and w in G is a subgraph of G defined by alternating sequence of distinct vertices and edges $v = v_0, e_0, v_1, e_1, \ldots, v_{p-1}, e_{p-1}, v_p = w$ such that every $e_i = \{v_i, v_{i+1}\} \in E$ for all i. Two vertices v and w are called *connected* in G, if there exists a path between v and w. A graph is called *connected* if every two vertices are connected, otherwise it is called *disconnected*. A *connected component* of a graph $G = (V, E)$ is the subset of vertices $S \subseteq V$ such that the induced subgraph $G[S]$ is connected and there is no edge between S and $V \setminus S$. The *distance* between two vertices v and w in G is the smallest length of a path from v to w. The largest distance in G is called the *diameter* of G and denoted by $diam(G)$. The *connectivity* $\kappa(G)$ of G is the minimum number of vertices whose deletion yields a disconnected graph or a graph with a single vertex. A *cycle* is the path with the same start and end vertices. The *density* of graph $G = (V, E)$ is the ratio of number of its edges to the total number of possible edges and denoted by $\rho(G) = \frac{|E|}{\binom{|V|}{2}}$.

For further information about graph related concepts and facts, we refer the reader to [7].

1.1.2 Graph families

Further in the dissertation, we will talk about graph clustering, hence a natural question arises: "What is a cluster?" There is no single answer to this question, as different applications may require different definitions of a "cluster". Typically they all require it to be some kind of a "tightly-knit" group of vertices. Here we will provide several models for a cluster as well as other graph types used throughout the study.

The most restricted model of a cluster is a complete graph, where every two vertices are adjacent (Figure 1.3), and its set of vertices is called a clique. That is, a clique is a subset of vertices

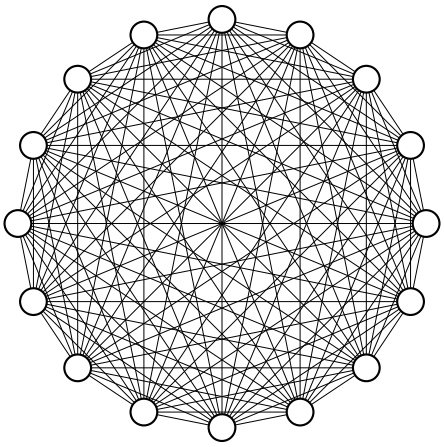


Figure 1.3: Complete graph on 16 vertices

inducing a subgraph which is a complete graph.

Definition 1. *A complete graph is a graph $G = (V, E)$ in which every two nodes are adjacent, i.e., for every $v, w \in V, v \neq w$ it holds that $\{v, w\} \in E$. A clique is a subset of vertices inducing a complete subgraph.*

However, for many application such definition of a cluster might be over restrictive. This issue can be addressed by relaxing certain clique defining properties [8]. The following proposition from aforementioned work suggests alternative definitions of a clique.

Proposition 1. *A subset of vertices C is a clique if and only if one of the following conditions*[3] *hold:*

1. $|E(G[C])| = \binom{|C|}{2}$;
2. $diam(G[C]) = 1$;
3. $\delta(G[C]) = |C| - 1$;
4. $\rho(G[C]) = 1$;
5. $\kappa(G[C]) = |C| - 1$.

[3]Not a complete list.

Here $|E(G[C])|$ denotes the set of edges of the induced subgraph $G[C]$.

"Relaxing", i.e., restrictively letting the right hand side to have different values, these defining properties help coming up with alternative definitions of a cluster. Consider the following examples.

Definition 2. *Given a fixed non-negative integer s, an s-defective clique is a subset of vertices $S \subseteq V$, such that*

$$|E(G[S])| \geq \binom{|S|}{2} - s.$$

In other words, it is a clique missing at most s edges.

Definition 3. *Given a fixed positive integer s, an s-plex is a subset of vertices $S \subseteq V$, such that*

$$\delta(G[S]) \geq |S| - s.$$

Definition 4. *Given a fixed $0 \leq \gamma \leq 1$, a γ-quasi clique is a subset of vertices $S \subseteq V$, such that*

$$\rho(G[S]) \geq \gamma.$$

Definition 5. *Given a fixed positive integer s, an s-bundle is a subset of vertices $S \subseteq V$, such that*

$$\kappa(G[S]) \geq |S| - s.$$

Definition 6. *Given a fixed positive integer s, an s-club is a subset of vertices $S \subseteq V$, such that*

$$diam(G[S]) \leq s.$$

An alternative way of defining clique relaxations is by enforcing a property that defines a clique on a set of vertices of fixed cardinality, e.g., $k + 1$ [8]. This leads to the following definitions.

Definition 7. *Given a fixed non-negative integer k, an k-core is a subset of vertices $S \subseteq V$, such*

that

$$\delta(G[S]) \geq k.$$

Definition 8. *Given a fixed non-negative integer k, an k-block is a subset of vertices $S \subseteq V$, such that*

$$\kappa(G[S]) \geq k.$$

As the reader may see, there are many ways to define a cluster using defining properties. As the list is not complete, and as different properties can be combined, we do not present all relaxations for the sake of brevity. We will only mention that there are other definitions of a cluster based on clique relaxations used in the literature, for example α-cluster [9] and k-community [10, 11].

In addition, there are certain graph families discussed later in the dissertation. Firstly, let's define a complement graph.

Definition 9. *A graph $G' = (V', E')$ is called a complement of $G = (V, E)$ if $V' = V$ and $E' = \binom{V}{2} \setminus E$.*

Based on the complimentarity principle, a well-studied structure is a stable or independent set.

Definition 10. *A stable or independent set is a subset of vertices in which every two vertices are not adjacent, i.e., the corresponding induced subgraph is the complement of a complete graph.*

Definition 11. *A bipartite graph is a graph $G = (V, E)$ whose vertices can be divided into two vertex-disjoint and independent sets U and V such that every edge connects a vertex in U to one in V.*

Definition 12. *A graph $G = (V, E)$ is a complete bipartite graph or biclique if V can be partitioned into two non-overlapping independent sets (it is bipartite) and every vertex of the first set is adjacent to every vertex of the second set.*

1.1.3 Graph clustering

As discussed before, it may be interesting and insightful to partition a network representation of a real-life system into "communities" or "tightly-knit" groups of nodes, either overlapping or

non-overlapping. The two main approaches presented in the literature are: 1) computing the partitioning/clustering based on some metric, e.g., modularity [12]; 2) partitioning the graph into clusters of a certain type, e.g., cliques, clique relaxations, k-communities, etc., and then trying either minimize the number of clusters or maximize the cluster coverage [8, 13]. In both approaches it is often important to know the initial number of clusters or seeds to complete the clustering process, as many algorithms assume it as their input. For example, one of the state of the art algorithms is called k-means clustering, which tries to cluster the data constructing groups of nodes that are "close" to each other, with a measure provided. Hence, we have particular interest in determining the "natural" number of clusters as the essential global characteristic of a network.

In order to estimate this number, we propose to focus on *cluster editing* [14] or *cluster vertex deletion* [15, 16] techniques. Suppose you are given the data with a constructed network representation which is perfectly clusterable. However, there are certain errors or noise in the inferred binary relation. There are two ways to "clean" the data, so that every connected component of a graph has a certain type, e.g., a clique. The first one is to edit, i.e., remove or add, the minimum amount of edges. This is a "liberal" approach as it keeps all data. The second, on the contrary, is the "conservative" one which tries to eliminate the minimum amount of vertices keeping only "good" data. In both approaches, the resulting number of clusters can be an input to other algorithms as well as clusters themselves can be used as "seed" clusters, i.e., those that can be expanded by assigning vertices from outside.

In Section 2 and Section 3 we study the structure called *independent union of cliques* (IUC). An IUC is graph whose every connected component is a clique, i.e., it is a vertex-disjoint union of cliques. We also generalize this problem for any other cluster type. The problem of finding the maximum IUC of a graph is relatively novel [1], and we analyze its behaviour on uniform random graphs and also propose an effective exact algorithm.

1.1.4 Optimization

Through the ages people tend to optimize the life around. Consider creation of a wheel to be able to transport heavy object efficiently all the way to updating our smartphones each year

for faster ones. Mathematical optimization roots from ancient time when about 300 B.C Euclid considers the minimal distance between a point and a line and shows that a rectangle with the greatest area is a square [17]. Centuries later, in 1657 Pierre de Fermat formulates his famous principle[4] that suggests nature behaves in the "optimal" way. Hence, by studying optimization, we also study the laws of the universe.

In this dissertation we consider mainly *combinatorial optimization* problems. This is a principle term without a rigorous definition. Generally speaking, combinatorial optimization deals with a finite collection of objects while looking for the "optimum" object. Instead of checking each object one by one, we attempt to incorporate more efficient methods [18].

Most of the problems discussed in this work ask to find the largest subset of vertices in a graph satisfying certain properties. For example, finding the largest cluster which is a clique is referred to as the maximum clique problem.

Definition 13. *Given a graph $G = (V, E)$ the maximum clique problem asks to find such subset of vertices $S \subseteq V$ such that $G[S]$ is a clique and $|S|$ is maximized.*

The size of a maximum clique is called the *clique number* of a graph G and denoted as $\omega(G)$. The size of a maximum independent set is called the *stability number* of a graph G and denoted as $\alpha(G)$. Other problems such as the maximum bipartite set, maximum s-plex, and others can be similarly defined.

1.1.5 Linear and Integer Programming

One of the most powerful approaches to solve a combinatorial optimization problem is to formulate it as an integer programming problem. There are powerful commercial software packages available solving such formulations "out-of-the-box". The scheme is the following: there are decisions to be made, each decision is represented by the value of one or more decision variables $x = \{x_i\}_{i=1}^{n}$; there is a way to measure which decision is better than the other by computing a numeric value, called the objective; additionally there are certain constraints to be met. Whenever the

[4] Or the principle of least time: a path taken between two points by a ray of light is the path that can be traversed in the least time.

objective and constraints are linear, such formulations are called *linear programming* (LP) models. A generic LP model can be described as

$$\text{(linear program)} \qquad \max_x \{c^T x \mid Ax \leq b\}.$$

The expression $c^T x$ is called the *objective function*. All decision satisfying constraints are *feasible* and altogether form the *feasible region* $\{x \mid Ax \leq b\}$. Linear problem is a subset of convex optimization and can be solved in polynomial time using ellipsoid method [19, 20]. Despite its theoretical quickness, this method typically demonstrates poor performance in practice and it is being replaced by efficient simplex method [21].

Even though LP models can simulate many processes and problems, they might be too restrictive for combinatorial optimization problems. Hence, decision variables are made to be restricted only to integer values, resulting in *integer linear programming* (IP) models, where the word "linear" is typically dropped. The problems with both integer and continuous decision variables are begin referred to as *mixed integer programming* (MIP) models. In special case decision variables can take only values 0 or 1, typically representing yes/no decision, which leads to a special class of 0-1 programming models. Lastly, for shortness, programming model is simply to be called a program.

$$\text{(integer program)} \qquad \max_x \{c^T x \mid Ax \leq b, x \in \mathbb{Z}^n\},$$

$$\text{(mixed integer program)} \qquad \max_x \{c^T x \mid Ax \leq b, x = (x_1, x_2), x_1 \in \mathbb{Z}^{n_1}, x_2 \in \mathbb{R}^{n_2}\},$$

$$\text{(0-1 program)} \qquad \max_{x \in \{0,1\}^n} \{c^T x \mid Ax \leq b\}.$$

A *linear relaxation* of an integer program $\max_x \{c^T x \mid Ax \leq b, x \in \mathbb{Z}^n\}$ is the linear program $\max_x \{c^T x \mid Ax \leq b\}$, i.e., where integral constraints are removed. Since integer programs are often $\mathcal{NP}$-hard, a linear relaxation is a fast way to find a bound on the optimal objective value.

For more information on integer programs please refer to [22, 23, 24].

1.2 Summary of Contributions

In Chapter 2, we define the maximum independent union of cliques (IUC), the maximum independent union of Π problems, and present their extensive analysis on uniform random graphs alongside other useful facts. These problems model clustering of a graph and thus are important to study from graph clustering perspective. One of the results is that the probability of a uniform random graph on n vertices with $p = 0.5$ form an IUC is $2^{-\binom{n}{2}}B_n$, where B_n is n-th Bell number, which nicely matches the probability $2^{-\binom{n}{2}}$ to form a clique; for other values of p the probability is unlikely to be expressed as a closed formula. We discover that the maximum IUC has a logarithmic upper bound on uniform random graphs. Furthermore, it is close to the maximum clique in dense graphs and far from the maximum independent set in sparse graphs. In addition, we show that every graph has an IUC at least logarithmic size, implying that uniform random graphs are the worst graphs to cluster in some sense. Hence, a reasonable definition of a cluster property Π may include the requirement to have a logarithmic upper bound on the size of a maximum IUΠ set in ER graphs. We provide easily verifiable sufficient conditions to detect such properties. After applying them for various clique relaxations, we discover that the independent union of s-defective cliques and s-plex satisfy the bound, the independent union of γ-quasi cliques satisfies the bound for sparse graphs and certain values of γ, while s-clubs do not satisfy this requirement and hence are not recommended to be used as a cluster model in this setting.

In Chapter 3, we study the Russian Doll Search (RDS) algorithm which can be efficiently used to find the maximum IUC in a graph as well as find other hereditary subgraphs. We present the key element of the algorithm, called a verifier, for IUC with linear complexity as well as for a biclique with constant complexity. Afterwards, we develop the parallel version of the algorithm for the shared-memory and distributed-memory computing models, perform extensive computational experiment to measure the speed up, and obtain best solutions known so far. It appears that the shared-memory version of the algorithm must perform not worse than the sequential one for unweighted graphs. This allows comparison of the combinatorial algorithms to IP solvers be more fair as the latter ones often efficiently utilize parallel capabilities of modern computers. We also

emphasize a certain algorithmic property which might help to develop faster verifiers in the future.

In Chapter 4, we provide the first study of the Hadwiger's number of a graph from optimization perspective. We provide several integer formulations as well as a heuristic and scale-reduction techniques, and compute the number alongside the well-known chromatic number, supporting the longstanding Hadwiger's conjecture. We discover that the separator-based formulation performs the best and establish an extended formulation for the problem.

Finally, we conclude and present future research ideas in Chapter 5.

2.1 Background and research questions

The maximum IUC problem, i.e., finding the subgraph with the largest amount of vertices which is an IUC, and the complementary problem, *cluster vertex deletion*, i.e., deleting the minimum amount of vertices so that the resulting graph is an IUC, received little attention in the literature. This problem is proved to be $\mathcal{NP}$-hard on various graph types, such as planar, claw-free, and bipartite, together with the first computational study [1]. Another direction of the research focuses on fixed-parameter tracking algorithms [15, 16].

In this section we study the maximum IUC problem as well as its generalization on uniform random graphs. We present the logarithmic upper bound, which suggests the quasi-polynomial (subexponential) algorithm to find the maximum IUC. On the one hand, this is somewhat unexpected, given that both cliques and independent sets yield feasible solutions for the corresponding problems and their independent union is suspected to be much larger, and they both have a logarithmic lower bound [25, 26]. On the other hand, this result is supported by the observation that uniform random graphs are "unclusterable" by design.

Uniform random graphs often appear as initial testing instances for new algorithms, thus it is important to know if the problem is "easy" and the performance gain with respect to previous algorithms happens because of exploiting characteristics of these graphs or actual improvements. In addition, uniform random graphs typically reveal useful insights about a problem. It is fair to note, that these graphs rarely appear as the result of modeling a real-life network. Those typically satisfy the small-world property and are power-law graphs [27].

2.2 Definitions and basic observations

2.2.1 Uniform random graphs

A *uniform random graph*, also called Erdős-Rényi (ER) graph, is a graph $G(n, p)$ with n vertices and where every edge has a constant probability $0 \leq p \leq 1$ to appear independently at

random [28, 29, 30]. The classical result by Matula [25] and later enhanced by Bollobas [26] implies, that the size of the maximum clique of a random graph $w(G(n,p))$ is at most $O(\log_{\frac{1}{p}}(n))$. Of course, this implies that the size of the maximum independent set $\alpha(G(n,p))$ is at most $O(\log_{\frac{1}{1-p}}(n))$. For more about structures in uniform random graphs, we refer the reader to [28, 31, 32].

2.2.2 Independent unions

Generalizing the IUC structure, we provide the definition about an independent component and independent union of properties. A property Π is an abstract description which selects certain graphs among all graphs, for example clique, 3-plex, bipartite, or any other.

Definition 14. *An independent component of graph $G = (V, E)$ is a subset of vertices $I \subseteq V$ such that there are no edges between I and $V \setminus I$.*

This definition is similar to the definition of a connected component in Section 1.1.1, relaxing the connectivity requirements.

Definition 15. *For a given property Π, a Π-set S is a graph/induced subgraph $G[S]$ satisfying Π.*

Definition 16. *An independent union of Π-sets (IUΠ) is a graph/induced subgraph $G[S]$, where exists partitioning such that every independent component is a Π-set and denoted as $\mathcal{U}(\Pi)$.*

It is important to define IUΠ set carefully. Definition using connected components fails when Π is disconnected; it is also not rigorous because partitioning a graph into independent components might not be unique. Consider a graph that is an independent set. Then any partitioning yields a union of independent components. Hence, we require only the existence of such partitioning. The maximum IUΠ problem asks to find the maximum induced $\mathcal{U}(\Pi)$ subgraph, its size is called the Π-*independence number* of graph G, and denoted as α^{Π}. This is the extension to the notation about IUC problem[1], where the maximum IUC size is denoted as α^{ω}.

Even though it may be difficult to verify quickly for certain Π if a graph is IUΠ, if Π is a clique, there is a simple verification for IUC. There is a well-know folklore criteria, proved, for example, here [1], that checks existence of open triangles, i.e., paths on three vertices (Figure 2.1).

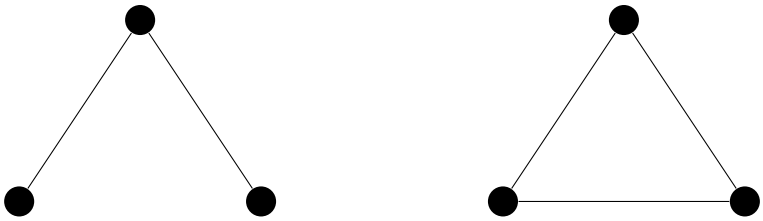


Figure 2.1: Open and closed triangles

Theorem 1. *A graph $G = (V, E)$ is an IUC if and only if it does not contain any open triangles as induced subgraphs.*

Proof. If a graph is an IUC and contains an open triangle then there are two nodes (endpoints of that open triangle) that are not in a clique but in the same connected component. Other way, if there are no open triangles, in a connected component every two nodes are adjacent and it is a clique. Indeed, if they are not, there still exists a path, since they share the same connected component (and we consider components with size at least three). Choose the shortest one, its length is at least three. Consider first three vertices. If they do not form an open triangle, that there is a path strictly shorter and this is a contradiction. □

2.2.3 Defining functions

For the results in this section, we require the concept of defining functions.

Definition 17. *Given a random graph $G(n, p)$ and a property Π, a defining function $\zeta(\Pi; n, p)$ is the probability that G satisfies Π.*

For rigorous usage of graph properties, we enforce certain constraints on them. We call a property *nontrivial*, if every graph with one vertex satisfies Π, and not every graph satisfies Π. We call a property *interesting*, if there are graphs with arbitrary large number of vertices satisfying Π. We call a property *connected*, if Π implies graph connectivity. An interesting fact is that if a property Π is also *hereditary on induced subgraphs*, i.e., for every $S \subseteq V$ such that $G[S]$ satisfies Π, for any nonempty $S' \subset S$ induced subgraph $G[S']$ satisfies Π, finding the maximum cardinality

Π	$\zeta(\Pi; n, p)$
clique	$p^{\binom{n}{2}}$
stable	$(1-p)^{\binom{n}{2}}$
s-defective	$\sum_{i=0}^{s} \binom{\binom{n}{2}}{i} p^{\binom{n}{2}-i}(1-p)^i$
γ-quasi	$\sum_{i=0}^{\lfloor(1-\gamma)\binom{n}{2}\rfloor} \binom{\binom{n}{2}}{i} p^{\binom{n}{2}-i}(1-p)^i$

Table 2.1: Examples of defining functions.

subset of vertices satisfying Π is $\mathcal{NP}$-hard [33]. Examples of hereditary properties are a clique, s-defective, and s-plex. Trivially, if Π is hereditary, then $\mathcal{U}(\Pi)$ is also hereditary. From now we assume that any considered property Π is nontrivial and interesting, unless mentioned otherwise.

Proposition 2. *Given any graph property Π, the following holds for its defining function:*

1. $\zeta(\mathcal{U}(\Pi); n, p) \geq \zeta(\Pi; n, p)$,
2. $\zeta(\Pi; 1, p) = 1$ *if* Π *is nontrivial,*
3. $\zeta(\mathcal{U}(\Pi); n, p) \geq (1-p)^{\binom{n}{2}}$ *if* Π *is nontrivial,*
4. $\zeta(\Pi; n, p) \leq \zeta(\Pi; n-1, p)$ *if* Π *is hereditary.*

Proof. 1. Π-set is also a $\mathcal{U}(\Pi)$-set.

2. Every singleton vertex satisfies any nontrivial property Π.

3. An independent set satisfies $\mathcal{U}(\Pi)$ for nontrivial Π.

4. Consider $G(n, p)$ and a graph after removal of a single vertex G'. Then $\mathbb{P}\{G \text{ satisfies } \Pi\} = \mathbb{P}\{(G \text{ satisfies } \Pi) \text{ AND } (G' \text{ satisfies } \Pi)\} \leq \mathbb{P}\{G' \text{ satisfies } \Pi\}$.

□

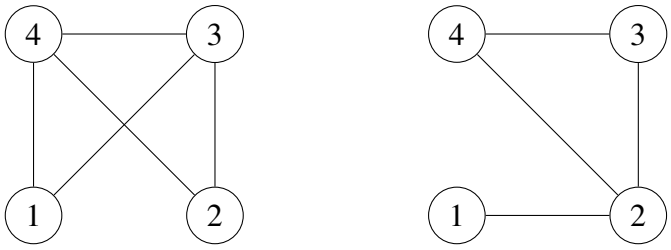


Figure 2.2: Proof for Observation 1

2.3 Open triangles in random graphs

Since open triangles (Figure 2.1) play a vital role in description of the IUC structure, it is interesting to study their behavior in general and on ER graphs. Following observations have been made.

Observation 1. *There are non-isomorphic graphs with identical open triangle sets.*

Proof. Refer to Figure 2.2. Open triangles are $\{1, 2, 3\}$ and $\{1, 2, 4\}$ for both graphs. □

Observation 1 tells that open triangle set does not define the initial graph. However, they still completely define the solution for the maximum IUC problem.

Observation 2. *The number of open triangles for a graph G with adjacency matrix A can be computed by formula*

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} (A^2 \circ (\mathbf{1} - A))_{ij},$$

where $A \circ B$ is a Hadamard (element-wise) matrix product and $\mathbf{1}$ is the matrix with all elements equal to 1.

Proof. By the simple counting argument. For details please refer to [34]. □

Observation 3. *The expected number of open triangles in a random graph $G(n; p)$ can be computed by the formula*

$$\binom{n}{3} 3p^2(1-p).$$

Proof. Either by substituting $A = \{a_{ij}\} = p$ in Observation 2 or noticing that a set of three vertices forms a open triangle with probability $3p^2(1-p)$. □

The Observation 3 suggests that graphs with density close to $\frac{2}{3}$ have the maximum number of open triangles.

The following natural question arises: what is the maximum number of open triangles in a graph? The very first guess is the number of all possible triplets (subsets size 3). However, as shown in [34], this is not true. The maximum number of open triangles in any graph G with n vertices is approximately $\frac{3}{4}\binom{n}{3}$ (Figure 2.3). We list the theorem here, however provide only the proof for even n, based on counting in [35]. For both odd and even n argument, please refer to the aforementioned work [34]. Counting various triangles is shown to be fruitful for the graph analysis toolkit [36, 37].

Theorem 2. *The maximum number of open triangles in a graph G with n vertices is given by the formula*

$$\binom{n}{3} - \binom{\lfloor \frac{n}{2} \rfloor}{3} - \binom{\lceil \frac{n}{2} \rceil}{3}, \tag{2.1}$$

and the extreme graph being a complete bipartite graph with balanced parts.

Proof. Consider a graph $G = (V, E)$ with n vertices and m edges. Let $A = \{a_{ij}\}_{\{i,j\} \in \binom{V}{2}}$ be the adjacency matrix. We introduce the notation

$$\Delta_i(G) = \{\{u, v, w\} : u, v, w \in V, a_{uv} + a_{uw} + a_{vw} = i\}, \quad i = 0, 1, 2, 3,$$

i.e., Δ_3 is a full triangle, Δ_2 is an open triangles, Δ_0 is a triplet without any edges, and finally Δ_1 is the only case left (partial triangle).

Every edge is part of some triangle (full, open or partial). For every edge, there are $n-2$ remaining vertices; with all of them it forms a triangle. Summing all up, full triangles are counted 3 times, open triangles 2 times, partial triangle once. Hence,

$$m(n-2) = 3|\Delta_3| + 2|\Delta_2| + |\Delta_1|. \tag{2.2}$$

Let x_j be the number of vertices of degree j in the graph G. Then

$$\sum_{j=0}^{n} x_j = n \quad \text{and} \quad \sum_{j=1}^{n} j x_j = 2m. \tag{2.3}$$

Consider a neighborhood of a vertex; every 2 edges from it are a part of either full or open triangle. Moreover, summing all up, full triangles are counted 3 times while open triangles only once. Hence,

$$\sum_{j=2}^{n} \binom{j}{2} x_j = 3|\Delta_3| + |\Delta_2|. \tag{2.4}$$

Subtracting (2.4) from (2.2), the number of open and partial triangles is

$$|\Delta_1| + |\Delta_2| = m(n-2) - \sum_{j=2}^{n} \binom{j}{2} x_j. \tag{2.5}$$

Setting equation (2.5) as the objective of an integer program (maximize $|\Delta_2| \leq |\Delta_1| + |\Delta_2|$) with constraints (2.3), then substituting m, the following IP is formed:

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} \left(\tfrac{n-2}{2} j - \tbinom{j}{2} \right) x_j \\ \text{s.t.} \quad & \sum_{j=0}^{n} x_j = n \\ & x_j \in \mathbb{Z}_+ \cup \{0\}, \quad j = 0, \ldots, n. \end{aligned} \tag{2.6}$$

This is a knapsack problem and its solution can be found analytically [22] by saturating the object with the most cost. The cost for x_j is $\frac{1}{2}\left(-j^2 + (n-1)j\right)$, parabola with one maximum located at $j^* = \frac{n-1}{2}$. If n is even, saturate either combination of x_{j_1} and x_{j_2}, where $j_1 = \lfloor \frac{n-1}{2} \rfloor$ and $j_2 = \lceil \frac{n-1}{2} \rceil$. Thence, the objective value equals to the one claimed in the theorem and achieved for a complete bipartite graph. For odd n this method results in a non-zero number of partial triangles and does not provide a tight bound. □

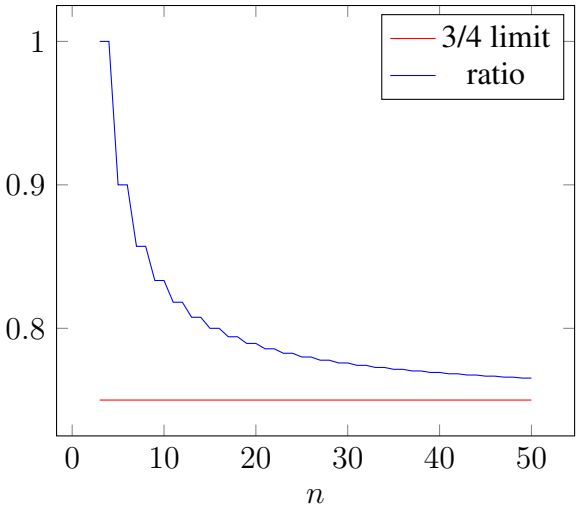


Figure 2.3: Ratio of the maximum number of open triangles to the number of all triplets

2.4 Random graph as critical graph

The maximum IUC problem has an interesting question: what is the smallest maximum IUC in a graph G with n vertices? A clique is a feasible solution and its smallest subgraph is just a singleton vertex, if G is an independent set; an independent set is also a feasible solution with the same smallest subgraph when G is a complete graph. However, these two observations does not provide any insights about IUC and it appears to be much more challenging question. For further results, we need several facts about Ramsey numbers theory [38].

Definition 18. *A Ramsey number $R(s,t)$ is the smallest number such that every graph $G = (V, E)$ with $|V| \geq R(s,t)$ must contain either a clique of cardinality s or an independent set of cardinality t.*

When a graph has at least $R(s,s)$ vertices, then its maximum IUC is at least s. The result from 1935 by Erdős and Szekeres [39] states[1] that

$$R(s,s) \leq \binom{2(s-2)}{s-2} + 1.$$

[1]Currently there exist better bounds [40, 41], however they do not significantly improve our result.

Applying Stirling approximation yields $R(s,s) \leq 4^{s-2}+1$. Hence, for a graph with n vertices, $\log_2(n-1) \leq 2(s-2)$, implying that the smallest maximum IUC is greater than $\log_2(n)/2$. This result also suggests that random graphs may be considered as "critical" graphs, where this estimate is tight with respect to a constant, as follows from our bound in Section 2.6.1.

2.5 Sufficient conditions for a logarithmic bound

Following the framework by Matula in [25] for the maximum clique problem in uniform random graphs, we extend it for any graph property using defining functions. Let X_k be a random variable representing the number of subgraphs with k vertices satisfying a property Π in $G(n,p)$. Then probability

$$\mathbb{P}(X_k \geq 1) \leq \mathbb{E}[X_k] = \binom{n}{k}\zeta(\Pi; k, p).$$

The idea is to find the largest k such that $\mathbb{E}[X_k]$ is still greater than 1. Conversely, given some function $f(n)$ where n is the number of vertices of a graph G, if $\mathbb{E}[X_{f(n)}] \to 0$ when $n \to \infty$, then G has no Π-set with high probability. The next lemma is the extension of the fact used in Matula's work combining with preceding ideas, substituting Π with $\mathcal{U}(\Pi)$.

Lemma 1. *Consider a uniform random graph $G(n,p)$ and a nontrivial, interesting property Π. Let*

$$\zeta(\mathcal{U}(\Pi); n, p) \leq a^{\binom{n}{2}} Q_n \tag{2.7}$$

for some $a \in (0,1)$ and $\{Q_n : n \geq 1\}$, such that

$$Q_{n+1} \leq CnQ_n \tag{2.8}$$

for some $C > 0$, and $n > N$, where N is some natural number. Then the Π-independence number $\alpha^{\Pi}(G(n,p))$ satisfies the inequality

$$\alpha^{\Pi}(G(n,p)) \leq (2+\epsilon)\left\lceil \log_{\frac{1}{a}}(n)\right\rceil + 1 \tag{2.9}$$

for any $\epsilon > 0$ with high probability.

Proof. From the recurrence relation for Q_n, it follows that $Q_n \leq C^{n-1}(n-1)!Q_N$. Then,

$$\zeta(\mathcal{U}(\Pi); n, p) \leq Q_N C^{n-1}(n-1)!a^{\binom{n}{2}}$$

and

$$\mathbb{E}[X_k] = \binom{n}{k}\zeta(\mathcal{U}(\Pi); k, p) \leq \binom{n}{k}Q_N C^{k-1}(k-1)!a^{\binom{k}{2}} \leq Q_N \frac{1}{Ck}\left(Cna^{\frac{k-1}{2}}\right)^k.$$

Substituting k with $f(n) = (2+\epsilon)\log_{\frac{1}{a}}(n) + 1$, we obtain

$$\mathbb{E}[X_{f(n)}] \leq Q_N \frac{1}{Cf(n)}\left(C \cdot n^{-\epsilon/2}\right)^{f(n)}.$$

The right-hand side of the last inequality tends to 0 when $n \to \infty$. Thus, the cardinality of a maximum $\mathcal{U}(\Pi)$-set is bounded above by $f(n)$ with high probability. □

Establishing logarithmic lower bounds in uniform random graphs is easy. Bollobas in [26] showed this for a clique and consecutively an independent set in a random graph. Hence, the following lemma holds.

Lemma 2. *For an interesting, nontrivial property Π, the $\mathcal{U}(\Pi)$-independence number of a uniform random graph $G(n,p)$ satisfies the inequality*

$$\alpha^{\Pi}(G(n,p)) \geq (1+\epsilon)\log_{\frac{1}{1-p}}(n)$$

for any $\epsilon > 0$ with high probability. Furthermore, if Π holds for complete graphs, then

$$\alpha^{\Pi}(G(n,p)) \geq (1+\epsilon)\log_{\frac{1}{p}}(n)$$

for any $\epsilon > 0$ with high probability.

Proof. If Π is trivial, then $\alpha^{\Pi}(G) \geq \alpha(G)$. From [26], $\alpha(G) \geq (1+\epsilon)\log_{\frac{1}{1-p}}(n)$.

If Π holds for complete graphs, then $\alpha^{\Pi}(G) \geq \omega(G)$. From [26], $\omega(G) \geq (1+\epsilon)\log_{\frac{1}{p}}(n)$. $\square$

Upper bounds are much more difficult to establish in practice. As it can be seen, in order to apply Lemma 1, one should be able to calculate or at least to estimate $\zeta(\mathcal{U}(\Pi); n, p)$ given a property Π; often there exists no closed formula. The next theorem demonstrates the full-history recurrent relation to serve this purpose.

Theorem 3. *Given a nontrivial, interesting property* Π *with the defining function* $\zeta(\Pi; n, p)$*, the defining function of* $\mathcal{U}(\Pi)$ *satisfies the recurrence relation*

$$\zeta(\mathcal{U}(\Pi); n, p) \leq \sum_{k=1}^{n} \binom{n-1}{k-1} \zeta(\Pi; k, p)(1-p)^{k(n-k)} \zeta(\mathcal{U}(\Pi); n-k, p), \tag{2.10}$$

where $\zeta(\mathcal{U}(\Pi); 0, p) = \zeta(\mathcal{U}(\Pi); 1, p) = 1$*. If* Π *is connected then* (2.10) *holds at equality.*

Proof. Assume Π is connected. Fix a vertex v_0. Let V denote the set of vertices of $G(n,p)$. If $G(n,p)$ satisfies $\mathcal{U}(\Pi)$ then the connected component of $G(n,p)$ that contains v_0 satisfies the property Π. Assume that this connected component has k vertices given by a Π-set S_k^{Π}. The $k-1$ vertices to form the set S_k^{Π} together with v_0 can be selected in $\binom{n-1}{k-1}$ different ways. Let

- $\mathbb{P}_1(n,k,p)$ denote the probability that S_k^{Π} is a Π-set in $G(n,p)$;
- $\mathbb{P}_2(n,k,p)$ be the probability that $V \setminus S_k^{\Pi}$ is a $\mathcal{U}(\Pi)$-set in $G(n,p)$;
- $\mathbb{P}_3(n,k,p)$ denote the probability that there are no edges between S_k^{Π} and $V \setminus S_k^{\Pi}$ in $G(n,p)$.

Then the defining function of $\mathcal{U}(\Pi)$ can be written as

$$\zeta(\mathcal{U}(\Pi); n, p) = \sum_{k=1}^{n} \binom{n-1}{k-1} \mathbb{P}_1(n,k,p)\mathbb{P}_2(n,k,p)\mathbb{P}_3(n,k,p).$$

We have (see Figure 2.4 for an illustration):

$$\mathbb{P}_1(n,k,p) = \zeta(\Pi; k, p),$$

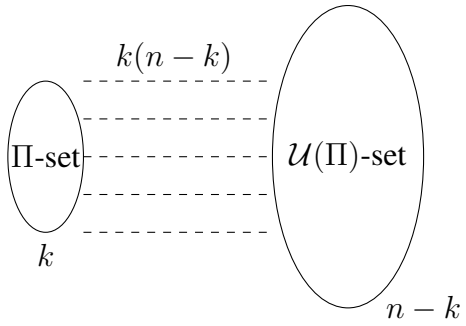


Figure 2.4: Illustration for the recurrence relation

$$\mathbb{P}_2(n,k,p) = \zeta(\mathcal{U}(\Pi); n-k, p),$$

$$\mathbb{P}_3(n,k,p) = (1-p)^{k(n-k)},$$

yielding (2.10) at equality. If Π is not connected, the equality transforms into inequality, because v_0 can belong to multiple k-vertex Π-sets. □

Corollary 1. *Inequality* (2.10) *can be rewritten as*

$$\zeta(\mathcal{U}(\Pi); n, p) \leq \sum_{i=0}^{n-1} \binom{n-1}{i} \zeta(\Pi; n-i, p)(1-p)^{i(n-i)} \zeta(\mathcal{U}(\Pi); i, p). \tag{2.11}$$

Observe that if we substitute $i(n-i) = \binom{n}{2} - \binom{i}{2} - \binom{n-i}{2}$ in (2.11), the resulting expression is a convolution of certain generating functions [42].

The next lemma introduces the technique that if a criterion for an overestimate holds for a connected Π, it will also hold for any Π.

Lemma 3. *Let*

$$\bar{\zeta}(\mathcal{U}(\Pi); n, p) = \sum_{k=1}^{n} \binom{n-1}{k-1} \zeta(\Pi; k, p)(1-p)^{k(n-k)} \bar{\zeta}(\mathcal{U}(\Pi); n-k, p), \tag{2.12}$$

where $\bar{\zeta}(\mathcal{U}(\Pi); 0, p) = \bar{\zeta}(\mathcal{U}(\Pi); 1, p) = 1$. *Then* $\bar{\zeta}(\mathcal{U}(\Pi); n, p) \geq \zeta(\mathcal{U}(\Pi); n, p)$ *for any* $n \geq 0$.

Proof. We use induction. The inequality clearly holds for $n = 0$. Assume it holds for any $n \leq N$.

Then

$$\begin{aligned}\bar{\zeta}(\mathcal{U}(\Pi);N+1,p) &= \sum_{k=1}^{N+1}\tbinom{N}{k-1}\zeta(\Pi;k,p)(1-p)^{k(N+1-k)}\bar{\zeta}(\mathcal{U}(\Pi);N+1-k,p)\\ &\geq \sum_{k=1}^{N+1}\tbinom{N}{k-1}\zeta(\Pi;k,p)(1-p)^{k(N+1-k)}\zeta(\mathcal{U}(\Pi);N+1-k,p)\\ &\geq \bar{\zeta}(\mathcal{U}(\Pi);N+1,p),\end{aligned}$$

where the first inequality is due to the induction assumption and the second inequality is due to (2.10). □

For two real-valued functions $f(x)$ and $g(x)$, we say that $f(x) = O(g(x))$ (pronounced as f is a big-O of g) if and only if there exists such $C > 0$ that starting from some x it holds that $|f(x)| \leq C|g(x)|$.

Now we are ready for the criteria for logarithmic upper bounds based on property Π.

Theorem 4. *The following asymptotic bounds on the* Π*-independence number* $\alpha^{\Pi}(G(n,p))$ *of a uniform random graph are valid.*

1. *If* $\zeta(\Pi;t+1,p)/\zeta(\Pi;t,p) = O(t(1-p)^t)$ *as* $t \to \infty$ *then*

$$\alpha^{\Pi}(G(n,p)) \leq (2+\epsilon)\left\lceil \log_{\frac{1}{1-p}}(n)\right\rceil + 1 \tag{2.13}$$

   with high probability for every $\epsilon > 0$.

2. *If* $p \geq 1/2$ *and* $\zeta(\Pi;t+1,p)/\zeta(\Pi;t,p) = O(tp^t)$ *as* $t \to \infty$ *then*

$$\alpha^{\Pi}(G(n,p)) \leq (2+\epsilon)\left\lceil \log_{\frac{1}{p}}(n)\right\rceil + 1 \tag{2.14}$$

   with high probability for every $\epsilon > 0$.

Proof. By Lemma 3, if we show that $\bar{\zeta}(\mathcal{U}(\Pi);n,p)$ satisfies (2.7) for some Q_n, so will $\zeta(\mathcal{U}(\Pi);n,p)$.

1. Denote by

$$Q'_n = (1-p)^{-\binom{n}{2}} \bar{\zeta}(\mathcal{U}(\Pi); n, p). \tag{2.15}$$

Then (2.7) is satisfied with $a = 1-p$ and $Q_n = Q'_n$, but (2.8) also needs to be ensured in order for Lemma 1 to apply. Equality (2.11) can be rewritten as

$$Q'_n = \sum_{i=0}^{n-1} \binom{n-1}{i} \zeta_1(n-i) Q'_i, \tag{2.16}$$

where

$$\zeta_1(n) = \frac{\zeta(\Pi; n, p)}{(1-p)^{\binom{n}{2}}}, \tag{2.17}$$

i.e., the ratio of the probability that $G(n,p)$ satisfies Π to the probability that graph's vertex set forms an independent set.

From (2.16) we obtain:

$$Q'_{n+1} = \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{n}{n-i} \zeta_1(n-i+1) Q'_i + Q'_n. \tag{2.18}$$

If $\zeta(\Pi; t+1, p)/\zeta(\Pi; t, p) = O(t(1-p)^t)$, then

$$\frac{1}{n-i} \zeta_1(n-i+1) \leq C' \zeta_1(n-i)$$

for some $C' > 0$. Hence,

$$Q'_{n+1} \leq (1 + C'n) Q'_n \leq 2C'n Q'_n$$

(assuming $C'n \geq 1$). Thus, (2.8) holds and the result follows immediately from Lemma 1.

2. Similarly, let

$$\bar{\zeta}(\mathcal{U}(\Pi); n, p) = p^{\binom{n}{2}} Q''_n. \tag{2.19}$$

Then (2.7) is satisfied with $a = p$ and $Q_n = Q''_n$. Equality (2.11) can be rewritten as

$$Q''_n = \sum_{i=0}^{n-1} \binom{n-1}{i} \zeta_2(n-i) \left(\frac{1-p}{p}\right)^{i(n-i)} Q''_i, \tag{2.20}$$

where

$$\zeta_2(n) = \frac{\zeta(\Pi; n, p)}{p^{\binom{n}{2}}}, \tag{2.21}$$

that is, the ratio of the probability that $G(n,p)$ satisfies Π to the probability that graph's vertex set forms a clique. Let $r = (1-p)/p$, then

$$Q''_{n+1} = \sum_{i=0}^{n-1} \binom{n-1}{i} \left(\frac{n}{n-i} \zeta_2(n-i+1) r^i\right) r^{i(n-i)} Q''_i + r^n Q''_n.$$

If $\zeta(\Pi; t+1, p)/\zeta(\Pi; t, p) = O(tp^t)$, then

$$\frac{1}{n-i} \zeta_2(n-i+1) r^i \le C'' \zeta_2(n-i)$$

for some $C'' > 0$. Hence,

$$Q''_{n+1} \le (1 + C''n) Q''_n \le 2C''n Q''_n$$

(assuming $C''n \ge 1$). Thus, (2.8) holds and the result follows immediately from Lemma 1. □

Corollary 2. *The results of Theorem 4 hold even when* $\zeta(\Pi; t+1, p)/\zeta(\Pi; t, p)$ *is replaced with* $\hat{\zeta}(\Pi; t+1, p)/\hat{\zeta}(\Pi; t, p)$*, where* $\hat{\zeta}(\Pi; t, p) \ge \zeta(\Pi; t, p)$ *for any* t*.*

Proof. Apply Theorem 4 with $\hat{\zeta}(\Pi; t, p)$ and notice that (2.7) holds true for $\zeta(\mathcal{U}(\Pi); n, p)$. □

Logarithmic upper bounds imply a quasi-exponential (subpolynomial) algorithm for finding the maximum IUΠ structure if $\mathcal{U}(\Pi)$-set is polynomially recognizable. Algorithm 1 checks all subsets with size less than specified upper bound $C\log(n)$ for n vertices; the running time is $O(2^{poly(C\log(n))})$, where $poly(n)$ is the time needed to recognize $\mathcal{U}(\Pi)$.

Algorithm 1 Find maximum IUΠ in $G(n,p) = (V,E)$

```
R = ∅
for every subset S ⊆ V such that |S| ≤ C log(|V|) do
    if |S| > |R| AND S satisfies 𝒰(Π) then
        R ← S
return R
```

2.6 Application to clique relaxations

In this section we provide the application our sufficient condition to clique relaxation such as s-defective clique, s-plex, and γ-quasi clique. Additionally, we present stronger results for the maximum IUC.

2.6.1 Clique

If property Π is a clique, then

$$\frac{\zeta(n+1,p)}{\zeta(n,p)} = p^n,$$

and conditions for Theorem 4 trivially hold.

Theorem 5. *The size of the maximum IUC in a uniform random graph $G(n,p)$ is at most $(2+\epsilon)\log_{1/p}(n)+1$ with high probability.*

However, there are much stronger and fruitful results for the maximum IUC in random graphs. The Table 2.2 illustrates the defining function of IUC for the first several values of n. When plotted versus the defining function of a clique, i.e., $\zeta(n,p) = p^{\binom{n}{2}}$, it is noticeable that for the dense case ($p > 0.5$) two functions converge (see Figure 2.5). This observation will be established formally in the following theorem.

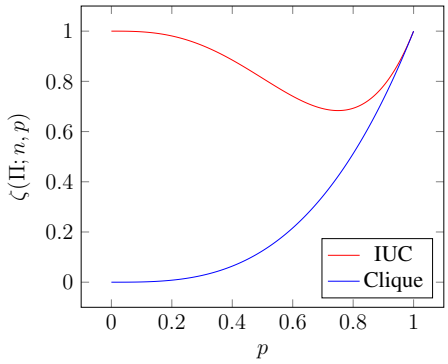
Theorem 6. *For $0.5 < p \leq 1$ we have $\zeta(IUC;n,p) \sim p^{\binom{n}{2}}$ as $n \to \infty$.*

Proof. We need to show that $\lim_{n\to\infty} \zeta(IUC;n,p)/p^{\binom{n}{2}} = 1$. Suppose

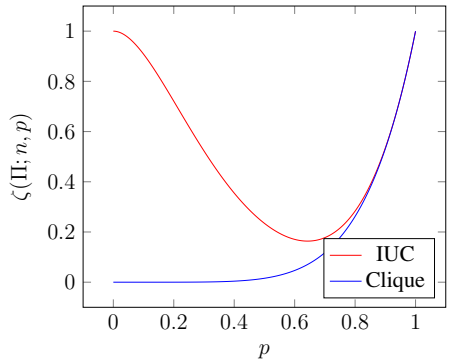
$$\zeta(IUC;t,p) = p^{\binom{t}{2}} Q_t,$$

n	$\zeta(IUC; n, p)$
0 – 2	1
3	$1 - 3p^2 + 3p^3$
4	$1 - 12p^2 + 32p^3 - 39p^4 + 24p^5 - 5p^6$
5	$1 - 30p^2 + 130p^3 - 270p^4 + 318p^5 - 195p^6 + 10p^7 + 75p^8 - 50p^9 + 12p^{10}$
6	$1 - 60p^2 + 360p^3 - 945p^4 + 792p^5 + 2775p^6 - 12090p^7 + 25035p^8 - 34240p^9$ $+33204p^{10} - 23130p^{11} + 11385p^{12} - 3780p^{13} + 765p^{14} - 71p^{15}$

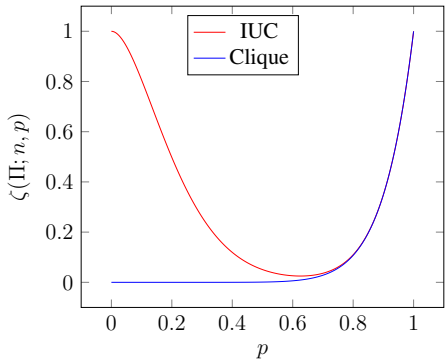
Table 2.2: $\zeta(n, p)$ for $n = 0, \ldots, 6$ for an IUC.



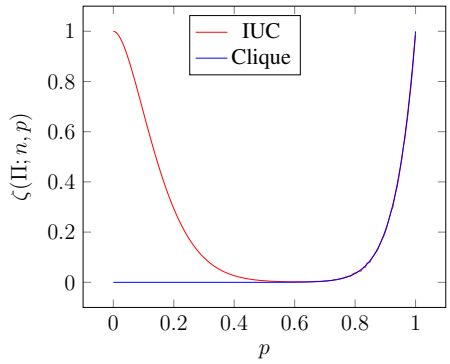
(a) $n = 3$



(b) $n = 4$



(c) $n = 5$



(d) $n = 6$

Figure 2.5: Probability $\zeta(\Pi; n, p)$ of the vertex set of $G(n, p)$ to form an IUC and a clique for $n = 3, 4, 5$, and 6

where $Q_0 = 1$. Then substituting to (2.11) we have

$$p^{\binom{n}{2}} Q_n = p^{\binom{n}{2}} \sum_{t=0}^{n-1} \binom{n-1}{t} \left(\frac{1-p}{p}\right)^{t(n-t)} Q_t. \tag{2.22}$$

Let $r = (1-p)/p$, then $0 \leq r < 1$. It is sufficient to prove that $Q_n \to 1$ when $n \to \infty$.

Note that

$$Q_n = 1 + \sum_{t=1}^{n-1} \binom{n-1}{t} r^{t(n-t)} Q_t.$$

Denoting $D_n = \sum_{t=1}^{n-1} \binom{n-1}{t} r^{t(n-t)} Q_t$, we obtain

$$D_n = \sum_{t=1}^{n-1} \binom{n-1}{t} r^{t(n-t)} \left(1 + D_t\right). \tag{2.23}$$

We need to show $D_n \to 0, n \to \infty$. Writing (2.23) for $n+1$ we have

$$D_{n+1} = \sum_{t=1}^{n-1} \binom{n-1}{t} \left(\frac{n}{n-t} r^t\right) r^{t(n-t)} \left(1 + D_t\right) + r^n \left(1 + D_n\right). \tag{2.24}$$

Observe that

$$\frac{n}{n-t} r^t \leq \frac{n}{n-1} r, \quad t = 1, \ldots, n-1. \tag{2.25}$$

Using (2.25) and factorizing (2.23) in (2.24) we can bound D_{n+1} as

$$D_{n+1} \leq \left(\frac{n}{n-1} r + r^n\right) D_n + r^n. \tag{2.26}$$

For $r < 1$ there exists b such that $r < b < 1$. The coefficient for D_n tends to r when $n \to \infty$, thus there exists a natural N such that for every $n \geq N$

$$D_{n+1} \leq b D_n + r^n. \tag{2.27}$$

Thus, we have

$$D_{n+m} \leq b^m D_n + r^n \sum_{i=0}^{m-1} b^i \leq b^m D_n + \frac{r^n}{1-b}. \tag{2.28}$$

Fix $\epsilon > 0$. Since $\lim_{n\to\infty} \frac{r^n}{1-b} = 0$, there exists $N_1 \in \mathbb{N}$ such that $\frac{r^n}{1-b} \leq \frac{\epsilon}{2}$ for every $n \geq N_1$. Also, $\lim_{m\to\infty} b^m D_{N_1} = 0$, so there exists such $N_2 \in \mathbb{N}$ such that $b^m D_{N_1} \leq \frac{\epsilon}{2}$ for every $m \geq N_2$. Therefore, for every $n \geq N_1 + N_2$ from (2.28) $D_n \leq \epsilon$. This proves that $D_n \to 0$ when $n \to \infty$. □

Theorem 6 implies logarithmic upper bound for an IUC using techniques from [25] similar to Lemma 1.

For the case when $p = 1/2$, a neat result can be shown. The defining function for an IUC convolutes into a closed formula which is similar to the defining function of a clique.

Theorem 7. *For $p = 0.5$, we have*

$$\zeta\left(IUC; n, 0.5\right) = 2^{-\binom{n}{2}} B_n, \tag{2.29}$$

where B_n is the n-th Bell number.

Proof. Here Π is a clique. Substitute $\zeta(\Pi; n, 0.5) = 2^{-\binom{n}{2}}$ to (2.11). Then

$$\zeta(\mathcal{U}(\Pi); n, 0.5) = 2^{-\binom{n}{2}} \sum_{t=0}^{n-1} \binom{n-1}{t} 2^{\binom{t}{2}} \zeta(\mathcal{U}(\Pi); t, 0.5). \tag{2.30}$$

Define $B_n = 2^{\binom{n}{2}} \zeta(\mathcal{U}(\Pi); n, 0.5)$. Then B_n satisfies the recursive relation

$$B_n = \sum_{t=0}^{n-1} \binom{n-1}{t} B_t \tag{2.31}$$

with $B_0 = 1$. This relation is precisely the relation for the n-th Bell number [42], so $\zeta(\mathcal{U}(\Pi); n, 0.5) = 2^{-\binom{n}{2}} B_n$. □

The result of Theorem 7 can be also obtained by a simpler thought argument. For the case $p = 1/2$, uniform random graphs $G(n, p)$ have a wonderful property that every graph has the equal

probability $2^{-\binom{n}{2}}$ to appear. Since the n-th Bells number B_n is the number of different partitions of a graph on n vertices [43], it is also the total number of IUCs on n vertices, since we can look at each partition as a clique and no edges between them. Hence, the probability of a graph to form an IUC is $2^{-\binom{n}{2}}B_n$.

Observation 4. *The number of graphs on n vertices that form IUC is B_n, where B_n is n-th Bell's number.*

Applying the bound (2.32) from [43]

$$B_n < \left(\frac{0.792n}{\ln(n+1)}\right)^n, \tag{2.32}$$

techniques from [25] similarly to Lemma 1 can be successfully applied independently from Theorem 4.

The last case is the sparse uniform random graph ($p < 0.5$). Here, even though for $p = 0$ the independent set is an IUC, for other $0 < p < 0.5$ it is apparently not a good approximation for an IUC as it follows from Theorem 8.

Theorem 8. *For $0 < p \leq 1$ we have $\zeta\left(IUC; n, p\right)/(1-p)^{\binom{n}{2}} \to \infty$ as $n \to \infty$.*

Proof. For $p > 0.5$ and $p = 0.5$ the result follows directly from Theorems 6 and 7, respectively. Let Π be a clique. Assume $0 < p < 0.5$. Let

$$\zeta(\mathcal{U}(\Pi); t, p) = (1-p)^{\binom{t}{2}}Q_t,$$

where $Q_0 = 1$. Then substituting this and $\zeta(\Pi; n-i, p) = p^{\binom{n-i}{2}}$ to (2.11) we obtain

$$(1-p)^{\binom{n}{2}}Q_n = \sum_{i=0}^{n-1}\binom{n-1}{i}p^{\binom{n-i}{2}}(1-p)^{\binom{n}{2}-\binom{n-i}{2}}Q_i.$$

Denoting $\bar{r} = p/(1-p)$, we get

$$(1-p)^{\binom{n}{2}} Q_n = (1-p)^{\binom{n}{2}} \sum_{i=0}^{n-1} \binom{n-1}{i} \bar{r}^{\binom{n-i}{2}} Q_i.$$

Thus, we need to prove that $Q_n \to \infty$ as $n \to \infty$ for

$$Q_n = \sum_{i=0}^{n-1} \binom{n-1}{i} \bar{r}^{\binom{n-i}{2}} Q_i.$$

For this purpose, we show that

$$Q_n \geq 1 + \binom{n}{2} \bar{r} \tag{2.33}$$

for any $n \geq 0$. We use induction on n. The statement clearly holds for $n = 0, 1$. Assume (2.33) holds for some $n \geq 2$. Then for Q_{n+1} we have

$$\begin{aligned} Q_{n+1} &= \sum_{i=0}^{n} \binom{n}{i} \bar{r}^{\binom{n+1-i}{2}} Q_i \\ &= Q_n + n\bar{r} Q_{n-1} + \sum_{i=0}^{n-2} \binom{n}{i} \bar{r}^{\binom{n+1-i}{2}} Q_i \\ &\geq 1 + \binom{n}{2} \bar{r} + n\bar{r} \left(1 + \binom{n-1}{2} \bar{r}\right) \\ &\geq 1 + \binom{n+1}{2} \bar{r}. \end{aligned}$$

This completes the proof. □

2.6.2 Defective clique

In this subsection we present the example of an application of Theorem 4 in the case when property Π is an s-defective clique.

Theorem 9. *The cardinality of a maximum $\mathcal{U}(\Pi)$-set in $G(n,p)$ is $O(\log n)$ with high probability when Π is an s-defective clique.*

Proof. We show that Theorem 4 applies in this case. We have

$$\zeta(\Pi^s, n, p) \leq \hat{\zeta}(\Pi^s, n, p) = \begin{pmatrix} \binom{n}{2} \\ s \end{pmatrix} p^{\binom{n}{2}-s}.$$

The fraction

$$\frac{\hat{\zeta}(\Pi^s, n+1, p)}{\hat{\zeta}(\Pi^s, n, p)} = p^n \frac{\begin{pmatrix} \binom{n+1}{2} \\ s \end{pmatrix}}{\begin{pmatrix} \binom{n}{2} \\ s \end{pmatrix}}.$$

We have

$$\frac{\begin{pmatrix} \binom{n+1}{2} \\ s \end{pmatrix}}{\begin{pmatrix} \binom{n}{2} \\ s \end{pmatrix}} = \frac{\left(\binom{n}{2}+n\right)!\left(\binom{n}{2}-s\right)!}{\left(\binom{n}{2}+n-s\right)!\binom{n}{2}!} = \prod_{i=0}^{s-1} \frac{\binom{n}{2}+n-i}{\binom{n}{2}-i}.$$

We can show that $\frac{\binom{n}{2}+n-i}{\binom{n}{2}-i} \leq 2$, $i \in \{0, \ldots, s-1\}$ for sufficiently large n. Thus,

$$\frac{\hat{\zeta}(\Pi^s, n+1, p)}{\hat{\zeta}(\Pi^s, n, p)} \leq 2^s n p^n$$

and the cardinality of a maximum independent union of s-defective cliques in $G(n,p)$ is $O(\log n)$ with high probability. □

2.6.3 Plex

In this subsection we present the example of an application of Theorem 4 in the case when property Π is an s-plex.

Theorem 10. *The cardinality of a maximum $\mathcal{U}(\Pi)$-set in $G(n,p)$, where $p < 0.5$, is $O(\log n)$ with high probability when Π is an s-plex.*

Proof. Observe that for an s-plex S the corresponding induced subgraph $G[S]$ has at most $\lfloor sn/2 \rfloor$ missing edges. Obviously, an s-plex is also an $\hat{s}$-defective clique with $\hat{s} = sn$. Hence,

$$\zeta(\Pi; n, p) \leq \hat{\zeta}(\Pi; n, p) = \binom{\binom{n}{2}}{sn} p^{\binom{n}{2}-sn}.$$

We have

$$\frac{\hat{\zeta}(\Pi; n+1, p)}{\hat{\zeta}(\Pi; n, p)(1-p)^n} = p^{-s} \left(\frac{p}{1-p}\right)^n \frac{\binom{\binom{n+1}{2}}{s(n+1)}}{\binom{\binom{n}{2}}{sn}}.$$

The fraction on the right can be rewritten as

$$\frac{\binom{\binom{n+1}{2}}{s(n+1)}}{\binom{\binom{n}{2}}{sn}} = \frac{\prod_{i=1-sn-s+n}^{n} \left(\binom{n}{2} + i\right)}{\prod_{i=1}^{s}(sn+i) \prod_{i=1-sn}^{0} \left(\binom{n}{2} + i\right)} = \frac{\prod_{i=1}^{n} \left(\binom{n}{2} + i\right)}{\prod_{i=1}^{s}(sn+i) \prod_{i=1-sn}^{n-sn-s} \left(\binom{n}{2} + i\right)}$$

$$\leq \frac{\prod_{i=1}^{n-s} \left(\binom{n}{2} + i\right) \prod_{i=n-s+1}^{n} \left(\binom{n}{2} + i\right)}{\prod_{i=1}^{n-s} \left(\binom{n}{2} - sn + i\right)} \leq \left(1 + \frac{3s}{n-s}\right)^{n-s} \prod_{i=1-s}^{0} \left(\binom{n}{2} + n + i\right)$$

$$\leq \exp\{3s\} \cdot f_s(n),$$

where $f_s(n) = \prod_{i=1-s}^{0} \left(\binom{n}{2} + n + i\right)$ is $O(n^{2s})$. Hence, the first condition in Theorem 4 holds for $p < 0.5$. □

2.6.4 Quasi clique

In this subsection we present the example of an application of Theorem 4 in the case when property Π is a γ-quasi clique.

Theorem 11. *The cardinality of a maximum $\mathcal{U}(\Pi)$-set in $G(n,p)$, where $p < 0.5$ and γ satisfies the inequality*

$$\frac{p^\gamma}{1-p} \cdot \frac{1}{\gamma^\gamma (1-\gamma)^{1-\gamma}} \leq 1, \tag{2.34}$$

is $O(\log n)$ with high probability when Π is a γ-quasi clique.

Proof. A γ-quasi clique can be seen as a defective clique, missing at most $\lfloor \gamma \binom{n}{2} \rfloor$ edges. Then the defining function satisfies

$$\zeta(\Pi, n, p) \leq \hat{\zeta}(\Pi, n, p) = \begin{pmatrix} \binom{n}{2} \\ \lfloor \gamma \binom{n}{2} \rfloor \end{pmatrix} p^{\lfloor \gamma \binom{n}{2} \rfloor}.$$

We have

$$\frac{\hat{\zeta}(\Pi, n+1, p)}{\hat{\zeta}(\Pi, n, p)(1-p)^n} \approx \left(\frac{p^\gamma}{1-p} \right)^n \frac{\begin{pmatrix} \binom{n+1}{2} \\ \lfloor \gamma \binom{n+1}{2} \rfloor \end{pmatrix}}{\begin{pmatrix} \binom{n}{2} \\ \lfloor \gamma \binom{n}{2} \rfloor \end{pmatrix}}. \tag{2.35}$$

By Stirling approximation,

$$\binom{k}{i} \sim \sqrt{\frac{k}{2\pi i (k-i)}} \frac{k^k}{i^i (k-i)^{(k-i)}}.$$

In particular, for $i = \lfloor \gamma k \rfloor$, we have $\binom{k}{\lfloor \gamma k \rfloor} \sim \left(\gamma^{\gamma k} (1-\gamma)^{(1-\gamma)k} \sqrt{2\pi \gamma (1-\gamma) k} \right)^{-1}$.

Hence, the binomial fraction in (2.35) can be estimated as

$$\left(\gamma^\gamma (1-\gamma)^{1-\gamma} \right)^{-n}.$$

The first condition in Theorem 4 is satisfied if (2.34) holds. □

Figure 2.6 shows the graphical region where inequality (2.34) holds.

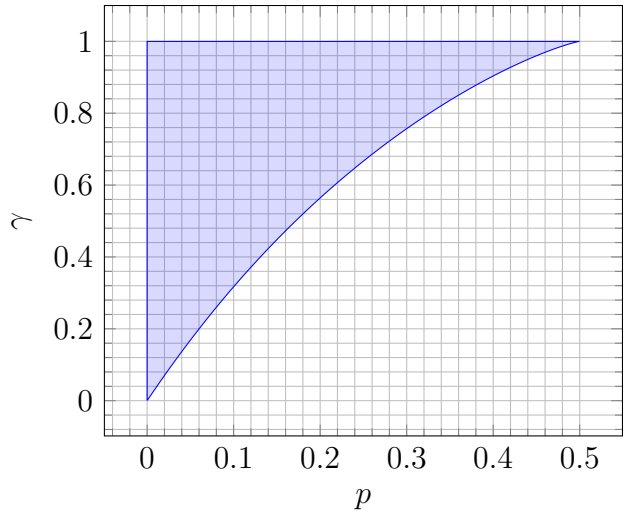


Figure 2.6: Bound region for γ-quasi clique in $G(n, p)$

2.6.5 Club

For $s = 1$, s-club is a clique. For $s > 1$, a star graph is an instance of an s-club. Thus, the node with maximum degree is a feasible solution to the maximum s-club problem. In a uniform random graph $G(n, p)$, the expected degree of any vertex is np [30], thus the size of the maximum s-club has no logarithmic upper bound. Trivially, an s-club is also a feasible solution to the maximum independent union of s-clubs problem.

2.7 Alternative approach

A slightly weaker bounds for certain clique relaxations may be obtained by using the following idea. Suppose for a fixed property Π, it is shown that every $IU\Pi$ size n must contain an IUC with size $\Upsilon(n)$. Then, if the maximum IUC is bounded by $F(n, p)$ in random graph $G(n, p)$, the size of the maximum $IU\Pi$ is bounded by $\Upsilon^{-1}(F(n, p))$ by a simple contradiction argument.

Definition 19. *A t-defective independent union of Π is a graph/subgraph where exists at most t edges whose deletion makes it an independent union of Π.*

Examples are a t-defective independent union of cliques, t-defective independent union of s-defective cliques, etc.

The following theorems are helpful to establish upper bounds for certain properties Π.

Theorem 12.

1. *Every s-defective clique C in G contains a clique with at least $|C| - s$ vertices.*

2. *Every s-plex C in G contains a clique with at least $\lceil |C|/s \rceil$ vertices.*

3. *Every γ-quasi clique C in G contains a clique with at least $\lceil |C|/(|C| - \gamma(|C|-1)) \rceil$ vertices.*

Proof. 1. Consider an s-defective clique C in G. Note that the graph $\bar{G}[C]$ has at most s edges, so its minimum vertex cover C' has at most s vertices. Then $C \setminus C'$ is a clique with at least $|C| - s$ vertices in G.

2. The second statement was proved in [44, Proposition 1]; here we offer a much simpler proof. According to Turán's theorem [45], we have $\omega(G) \geq \frac{|V|^2}{|V|^2 - 2|E|}$ for any simple, undirected graph $G = (V, E)$. Applying this theorem for $G[C]$ and taking into account that the number of edges in $G[C]$ is at least $|C|(|C| - s)/2$, we obtain $\omega(G[C]) \geq |C|/s$. Even simpler proof is an immediate result of Caro-Wei bound [46, 47, 48].

3. The result follows from Turán's theorem by observing that $G[C]$ has at least $\gamma|C|(|C| - 1)$ edges.

□

Theorem 13. *1. Let Π^s: the set of vertices is an s-defective clique. Then*

$$\alpha^{\Pi^s}(G) \leq \lfloor (1 + s/2)\alpha^{\omega}(G) \rfloor. \tag{2.36}$$

2. Let $\Pi^{s'}$: the set of vertices is an s-plex. Then

$$\alpha^{\Pi^{s'}}(G) \leq s\alpha^{\omega}(G). \tag{2.37}$$

3. *Let $\mathcal{U}_t(\Pi)$: the set of vertices is a t-defective independent union of Π-sets and Π is hereditary. Then*

$$\max |\mathcal{U}_t(\Pi)(G)| \leq \alpha^{\Pi} + t. \tag{2.38}$$

Proof. 1. Let U be a maximum independent union of s-defective cliques. We construct an IUC C by finding a maximum clique of each connected component of $G[U]$ and including it in C. If a connected component has 1 or 2 vertices, its whole vertex set is included in C. Otherwise, if a connected component has $k > 2$ vertices and $s' \leq s$ missing edges, its maximum clique has at least $r = \max\{2, k - s\}$ vertices by Theorem 12. The fraction of vertices from this connected component included in C is at least $r/(r + s) \geq 2/(2 + s)$. Thus, overall, at least $2|U|/(2 + s)$ of vertices from U are included in C.

2. Let U be a maximum independent union of s-plexes. We construct an IUC C consisting of maximum cliques of connected components of $G[U]$, each of which corresponds to an s-plex. According to Theorem 12, a connected component of $G[U]$ on k vertices contains a clique with at least k/s vertices. Thus, C must have at least $|U|/s$ vertices.

3. Immediately from its definition after removal endpoints of certain edges.

□

Note that for the property Π^{γ} requiring the set of vertices to be a γ-quasi clique, establishing a meaningful relation between $\alpha^{\Pi^{\gamma}}(G)$ and $\alpha^{\omega}(G)$ for arbitrary $\gamma \in (0, 1)$ using approach described in Theorem 13 appears to be more challenging.

3. PARALLEL RUSSIAN DOLL SEARCH FOR MAXIMUM HEREDITARY SUBGRAPH PROBLEM

3.1 Background and research questions

In Chapter 2 we studied the maximum IUC problem and its generalizations in uniform random graphs. However, the question of solving it on an arbitrary graph remains unanswered. In this chapter we study the Russian Doll Search (RDS) algorithm which can be used to solve the maximum weighted Π-set problem, if property Π is nontrivial, interesting, and hereditary on induced subgraphs. We develop a parallel version of the algorithm and demonstrate its performance for the following properties: IUC, biclique, s-plex, and s-defective clique.

In 2002, Östergård enhanced the enumerative maximum clique algorithm [49] by introducing certain smart prunes [50]. His version considerably outperformed existing approaches, especially for sparse graphs. Later the Östergård's approach was extended to the problem of finding a maximum hereditary subgraph, first in [51], and then in [2], which improves upon the results reported in [51]. We further enhance the algorithm by introducing its parallel versions and solving previously unsolved problem instances to optimality. It should be noted that the RDS approach was also used for other combinatorial problems; we refer the reader to [52, 53, 54, 55, 56, 57] for more information.

The RDS algorithm is sequential, however it may be beneficial to incorporate parallel capabilities of modern computers [58]. The books [59, 60, 61, 62] provide a solid foundation on parallel computing technologies and principles. For the algorithmic aspects, we refer the reader to [63, 64, 65]. In this work we use both classical frameworks in parallel computing, namely, the *shared-memory* and *distributed-memory* frameworks. The first one is often referred as a PRAM model [66] and is typically implemented by Open Multi-Processing or OpenMP [67] application interface. In this framework it is assumed that there are distinct threads running in parallel, which can read/write the same memory, typically taking advantage of the modern multi-core

CPUs [58, 68, 69]. The distributed-memory framework assumes there are distinct nodes with their own computational resources (memory, CPU) running in parallel. Those nodes can communicate between themselves in order to exchange information. This is typically done by setting up a supercomputer or a cluster with multiple processors/nodes and low-latency bridges in between. There might be multiple communication protocols, e.g. Parallel Virtual Machine (PVM) [70] or Message-Passing Interface (MPI) [71]. The latter one is typically thought to be the standard for the industry because of its efficiency and speed for tightly-coupled large-scale parallel systems [58]. It is also widely used in the current research in cluster computing [72, 73, 74]. More features and implementation details can be found in [69, 75]. It is worth mentioning that there are also GPU-based opportunities for parallel computing, which were used to solve, e.g., the maximum clique problem [76] and the maximum s-plex problem [77].

Computational results for integer programming approaches using GUROBI or CPLEX reported in the literature often benefit from parallelization implicitly used by these solvers. To make comparison of these approaches to RDS more fair, we propose parallel versions of RDS algorithm, which dramatically improves the running time for many instances.

The main difficulty with parallelizing RDS is preserving branch-and-bound prunes between parallel tasks. In this chapter, we prove that our shared-memory version performs no worse than the original algorithm for unweighted graphs, excluding the overhead needed to "parallelize", i.e., start running parallel tasks. The distributed-memory version has no such guarantee.

3.2 Main concepts

The high-performance computing systems nowadays allow a huge degree of parallelization. They can be loosely classified into shared-memory systems, distributed-memory systems, and grids [58]. A shared-memory system is usually represented by one or more central processing units (CPU), tightly coupled. Since processors are hardware-wired, they share the data access at memory speeds. Computations can be naturally synchronized to access the information, e.g., lock-down on write, thus the computation model for a shared-memory system is exactly the Parallel Random Access Memory (PRAM) model, studied, e.g., in [65]. For our purposes, we assume

that we are able to execute a certain number of threads, usually equal to the number of available cores in a multiprocessor, with automatic handling of concurrent memory accesses. One of the industry standards for creating and executing such threads is OpenMP [67]. It requires compiler as well as runtime support. There are also other possibilities for computational parallelism (see [58]). A supercomputer or a cluster is usually a collection of nodes, each of which has its own central processing unit (CPU), memory, as well as other resources. Such nodes can efficiently communicate between themselves, forming a distributed-memory system. Interestingly, each node still might have a multiprocessor as a CPU, allowing even further parallelization. The advantage of distributed memory systems is a considerably larger number of parallel steps that could be performed, as well as the total memory. For our purposes, a distributed-memory system is a collection of nodes executing the same code in parallel and allowing efficient communication, or messaging, among them. To be more precise, messaging techniques might still have certain limitations [58]. We use the Message-Passing Interface (MPI) [71] as our programming model. A cluster or every node of a supercomputer might have a multiprocessor, allowing parallelization being done using distributed-memory model between nodes first, and then using shared-memory model inside each node afterwards. We refer to such approach as a *hybrid* model.

3.2.1 Russian Doll Search and basic observations

The main idea of Russian Doll Search is to firstly solve the problem on small subgraphs, then, using the heredity property, effectively prune some computational branches. Suppose we are given a simple, undirected, and vertex-weighted graph $G = (V, E)$ with vertices $V = \{v_1, v_2, \ldots, v_n\}$. Define $G_i = G[\{v_i, v_{i+1}, \ldots, v_n\}]$, the "tail" subgraphs. Algorithm 2 iteratively finds optimal solutions $\mu[i]$ for induced subgraphs G_i and stores them. Also, the general lower bound is maintained. At each branching we have a candidate set C, the vertices that might be in the optimal solution, and a current solution set P we are trying to enlarge. This is a standard combinatorial branch-and-bound framework (see, e.g., [78]). Prune 1 is a typical pruning which stops looking for a better solution whenever the total weight of $P \cup C$ is less than the best solution found so far, as even if all the candidate vertices are added, the resulting set will still be suboptimal. Prune 2 is specific

to the RDS algorithm. In this step, if $i = \min_j\{v_j \in C\}$, then trivially $C \subseteq G_i$, and thus the best possible subgraph in C satisfying the given property Π is at most $\mu[i]$. Russian Doll Search algorithm can alternatively be viewed as a combination of depth and breadth searches, where the first cycle moves into depth, while inside each depth step we perform breadth search to find the best solution.

We are also using the idea proposed in [2] to maintain auxiliary information for each set pair (C, P) in order to improve the speed of verifying that a subgraph satisfies property Π. We add Step #23 in Algorithm 2 to allow the backtracking trick. This typically reduces the data structure sizes mentioned in [2] from quadratic to linear size, as instead of maintaining information for every recursive call, we are allowed to backtrack (revert) changes done by `Prepare_AUX` routine.

Algorithm 2 Russian Doll Search (RDS) for the Maximum Weight Π Problem

1: **procedure** RDS-ALGORITHM($G = (V, E)$, Π)
2: Order(V)
3: LB = 0
4: **for** $i = n - 1, \ldots, 0$ **do**
5: $C = \{v : (v, v_j)$ satisfies Π for $j = i + 1, \ldots, n\}$ ▷ Π verification
6: INIT_AUX(C, v_i)
7: FIND_MAX($C, \{v_i\}$)
8: $\mu[i] \leftarrow$ LB
9: FREE_AUX
10: **return** $\mu[0]$
11: **procedure** FIND_MAX(C,P)
12: **if** $C = \emptyset$ **then**
13: LB = $\max\{\text{weight}(P), \text{LB}\}$
14: **while** $C \neq \emptyset$ **do**
15: **if** weight(C) + weight(P) $\leq$ LB **then return** ▷ Prune 1
16: $i \leftarrow \arg\min_j\{v_j \in C\}$
17: **if** $\mu[i]$ + weight(P) $\leq LB$ **then return** ▷ Prune 2
18: $C \leftarrow C \setminus \{v_i\}$
19: PREPARE_AUX(C, P, v_i)
20: $P' \leftarrow P \cup \{v_i\}$
21: $C' = \{v \in C : P' \cup \{v\}$ satisfies $\Pi\}$ ▷ Π verification
22: FIND_MAX(C', P')
23: UNDO_AUX(C, P, v_i)

Given a certain property Π, the verification procedure for Step #21 needs to be specified. Its template is demonstrated in Algorithm 3; the goal is to answer if, given Π-set P and a vertex v, their union $P \cup \{v\}$ satisfies Π.

Algorithm 3 Generic verification procedure

```
1: procedure Π-VERIFIER(P,v)
2:     return P ∪ {v} satisfies Π
```

For example, if Π is a clique, its verification procedure might look as illustrated in Algorithm 4.

Algorithm 4 Bad clique verifier

```
1: procedure IS_CLIQUE(P,v)
2:     for every w ∈ P do
3:         if {w, v} ∉ E then
4:             return false
5:     return true
```

Observation 5. *The Π-verification step does not require verifying all sets $P \cup \{v\} \setminus \{u\}$, where P is the solution set, v is the candidate node to enter, and u is the last node that was added to P.*

Proof. The node v comes from a candidate set, which was filtered with respect to the set $P \setminus \{u\}$. □

Observation 5 simply states that if u is the last vertex added to the solution set P and v is from the candidate set C, then $P \setminus \{u\} \cup \{v\}$ is a Π-set. This is extremely important if a verifier is thought of in terms of forbidden subgraphs. Instead of testing every subset with v, one can check only subsets including both v and u. For example, if one thinks that a clique is a subgraph which does not have the forbidden structure consisting of two nonadjacent vertices, a clique verifier is simply checking if $\{u, v\} \in E$ (see Algorithm 5). We noticed that this is the verifier used in [50], however this observation was not carried over to new verifiers for hereditary properties.

Algorithm 5 Good clique verifier

```
1: procedure IS_CLIQUE(P,v)
2:     Let u be the last vertex added to P
3:     return {u, v} ∈ E
```

3.2.2 Enhanced verifier

Based on Observation 5, we enhance IUC verifier from [1] and present a constant time verifier for biclique subgraphs. In this section we represent the solution set P as the pair of lastly added vertex u and the remaining set $P' = P \setminus \{u\}$.

The verifier from [1] checks if every triplet of vertices forms an open triangle, as follows from Theorem 1. It runs in $O(|P|^2)$ time (see Algorithm 6). Applying Observation 5 and an open triangle being the forbidden subgraph, we can iterate only once through vertices in P', making the check in linear time (see Algorithm 7).

Algorithm 6 IUC verifier in [1]

```
1: procedure IS_IUC(P,v)
2:     for w ∈ P, u ∈ P do
3:         if 𝕀({u, v} ∈ E) + 𝕀({w, v} ∈ E) + 𝕀({u, w} ∈ E) = 2 then      ▷ indicator func. 𝕀
4:             return false
5:     return true
```

Algorithm 7 Enhanced IUC verifier

```
1: procedure IS_IUC(P = (P', u),v)
2:     for w ∈ P' do
3:         if 𝕀({u, v} ∈ E) + 𝕀({w, v} ∈ E) + 𝕀({u, w} ∈ E) = 2 then      ▷ indicator func. 𝕀
4:             return false
5:     return true
```

The idea of Observation 5 results in a constant time verifier when property Π is biclique, i.e.,

complete bipartite subgraph. Consider P', which is a biclique. It can be partitioned into two independent sets L' and R', and every vertex from one set is adjacent to all vertices in the other. Sets $P' \cup \{v\}$ and $P' \cup \{p\}$ are also bicliques. Conclusively, we need to check the following conditions. Assume that WLOG the first vertex w in P' belongs to L'. Then, checking if a vertex t is in L' or R' is simply testing if $\{w, t\} \in E$. Thus, if the last vertex u and candidate v belong to the same partition, then it must hold that $\{u, v\} \notin E$; otherwise, if they belong to different partitions, it must hold that $\{u, v\} \in E$. This logic translates into two XOR operations $\oplus$ and one negation operation $\neg$, as illustrated in Algorithm 8.

Algorithm 8 Biclique verifier

1: **procedure** IS_BICLIQUE($G = (V, E)$,$P = (P', u)$,v)
2: Let w be any vertex (e.g. the first) in P'
3: **return** $\neg\left((\{w, u\} \in E) \oplus (\{w, v\} \in E) \oplus (\{u, v\} \in E)\right)$

3.3 Shared-memory modification

The first parallelization we develop aims to distribute the cycle in Step #14 in Algorithm 2 between processor cores, i.e., the so-called breadth-search parallelism. This is a shared-memory implementation using OpenMP. The lower bound LB is shared between cores. Auxiliary information for routines `Prepare_AUX` and `Undo_AUX` is created for each core independently. Notice that all threads work for the same $\mu[i]$. Hence, it is easy to verify prunes correctness: prune 1 depends on the correct lower bound for G_i; prune 2 uses any lower bound LB for G_i and $\mu[i]$ for previously computed tail-graphs, however, these are not changed by parallel processes. The main challenge of this implementation is the efficient memory management for sets and auxiliary structures.

For breadth-search, we do only the first level of branching in parallel using a so-called dynamic scheduling. In this concept the next available task is assigned to a free thread in orderly fashion. Assume that the candidate set C is sorted in ascending order, allowing for efficient minimum

operations. As the rest of the computational tree depends on the size of the candidate set which is a part of G_i, the typical situation is that the candidate set diminishes with time, so it is crucial not to split it into consecutive parts per thread, as most of the work will be concentrated in the first chunk. The last technical detail includes storing the candidate set C as an array while keeping the starting index, which also points to the minimum element.

Algorithm 9 OpenMP parallelization

```
1:  procedure FIND_MAX_LEVEL1(C,P,LB)
2:      Sort C ascending
3:      Parallelize, assign thread number to t_i and number of threads to n
4:      for c_i = t_i with update c_i = c_i + n  do          ▷ Dynamic Scheduling
5:          i ← C[c_i]                                      ▷ Get minimum
6:          PREPARE_AUX(C, P, v_i)                          ▷ Per thread
7:          P' ← P ∪ {v_i}
8:          C' = {v ∈ C[c_i, ...] : P' ∪ {v} satisfies Π}   ▷ Π verification
9:          FIND_MAX(C', P')                                ▷ Branching sequentially, share LB
10:         UNDO_AUX(C, P, v_i)
```

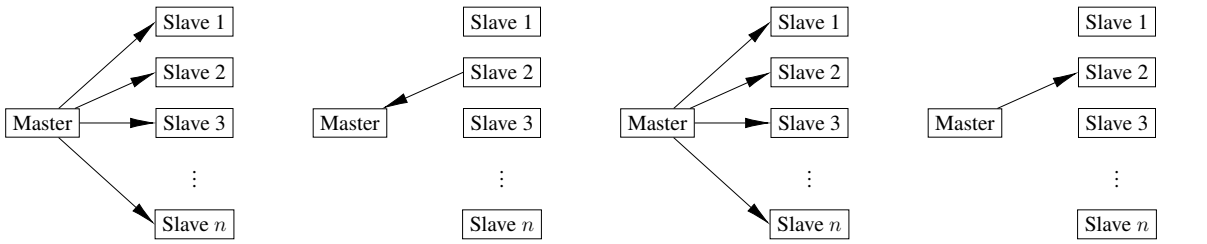
Theorem 14. *For the maximum* Π *problem (unweighted), Algorithm 9 has the same strength prunes as Algorithm 2. Hence, the running time of shared-memory version is at most the running time of sequential version plus the time needed to parallelize (create parallel tasks).*

Proof. Notice that if the problem is unweighted, when a better solution is found then $\mu[i] = \mu[i+1]+1$, otherwise $\mu[i] = \mu[i+1]$. Hence, both algorithms can stop computing $\mu[i]$ if LB was updated [50]. As Algorithm 9 does not change previous $\mu[i]$ values, all prunes checks have exactly the same LB and $\mu[i]$, and consequently have the same strength. □

3.4 Distributed-memory modification

A distributed-memory system consists of a collection of nodes with their own multiprocessors and memory resources, connected via low-latency network. A classical master-slave (primary-replica, primary-secondary, coordinator-worker) scheme is the following: exactly one of the nodes

is declared to be the *master*, and other nodes are named as *slaves*. The master splits the main problem into subproblems, assigns tasks to slaves, collects the results, and sends updates of problem parameters, e.g., lower bound. The typical sequence of events can be seen in Figure 3.1.



(a) Send initial tasks. (b) Receive a result. (c) Send updates. (d) Send the next task.

Figure 3.1: Master-Slave event sequence

For parallelizing RDS algorithm with distributed-memory nodes, we compute different $\mu[i]$ simultaneously (see step #4 in Algorithm 2), i.e., the master sends tasks to compute a certain $\mu[i]$. Each slave then runs Algorithm 9 and returns its LB. However, this imposes several problems. Both prunes might become incorrect: the lower bound LB is assumed to be the bound for the graph G_i when computing $\mu[i]$; some $\mu[i]$ values might not be available. The naive solution is to ignore prunes with uncomputed $\mu[i]$ and make master send updates whenever one of the slaves returns an updated value. However, this approach is not valid. Consider the following case: for some i^* it holds that $\mu[i^*] = 10$. Slave, computing $\mu[i^* + 2]$, finds the value 10 and returns it to the master, who sends updates to all slaves and they start performing prunes with this value. Next, another slave, computing $\mu[i^* + 1]$, finds the value 11. This implies that in sequential version $\mu[i^* + 2] = 11$, and previously performed prunes may have been incorrect. Hence, we can use only those $\mu[j]$ that have known $\mu[i]$ for all $i \in \{j, \ldots, n\}$. We propose the following modifications to address potential problems.

For the simplicity of implementation and arguments, assume that when a slave computes value of $\mu[i]$, the master sends its negation. When a slave is sure that this value can be used in the prunes, it is being recovered to a positive number. Hence, all $\mu[i]$ can be split into three groups:

1. if $\mu[i] > 0$ then it can be safely used in prunes;

2. if $\mu[i] = 0$ then it is uncomputed;

3. if $\mu[i] < 0$ then it was computed, and the value found equals $-\mu[i]$.

Using this idea, each slave maintains its own lower bound LB. While working on the task of computing $\mu[i]$, the lower bound LB(G_i) can be calculated as

$$LB(G_i) = \max_{i<j} |\mu[j]|,$$

because the lower bound is the best solution found in the graph G_i; even if some $\mu[j]$ might not be the best bound in G_j, their values still correspond to a certain feasible subgraph size. Every time a slave receives an update from the master, it attempts to restore some $\mu[i]$ values using Algorithm 10 and recomputes the lower bound. This algorithm makes $\mu[i]$ marked as "suitable for prunes" if and only if all previous $\mu[j]$ are computed for $j > i$.

Algorithm 10 Restore valid $\mu[i]$

```
1: procedure RESTORE(array μ[i])
2:     for i = {n, ..., 1} do
3:         if all μ[j] > 0 for all j ∈ {n, ..., i} AND μ[i] < 0 then   ▷ Check all previous
4:             μ[i] = −μ[i]                                            ▷ Restore
```

3.5 Computational analysis

All tests were performed on the computer with Intel® Xeon® CPU at 2.40GHz, 8 GB RAM, 16 cores. The code was compiled with GNU C++ compiler version 4.9.2. For the distributed-memory version, the cluster with four nodes identical to this machine was created and MPI version was MPICH 3.2. Complete set of results can be found in Appendix A. In this section we provide their summary and analysis.

Figure 3.2 illustrates speedup versus sequential Algorithm 2 for Algorithm 9 for the maximum IUC problem on selected instances. If T_{seq} is the time of the former and $T_{par}(k)$ is the time of the latter algorithms with k cores, then speedup S can be calculated as

$$S = \frac{T_{seq}}{T_{par}(k)}.$$

It can be clearly seen that for most instances there is a speedup which fits into Amdahl's law [79] scheme: more parallelizm provides faster performance, but at some point it takes too much time to parallelize and communicate between tasks. There are some instances that have constant speedups after a certain number of cores is in use. This happens because `find_max` recursion consistently updates the lower bound LB in the first parallelization chunk. We were able to locate an instance where the shared-memory algorithm could not achieve any speedup. This happens when the solution to the problem is relatively large and the algorithm is able to update the lower bound while working on the beginning of initial candidate set.

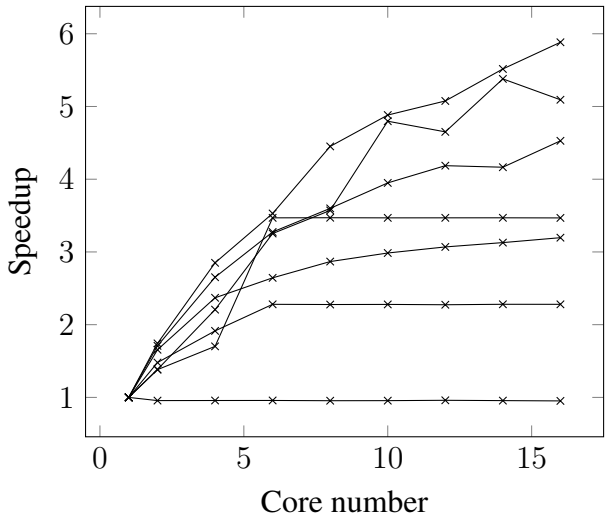


Figure 3.2: Speedup versus core number for IUC problem for selected graphs

The next round of experiments was conducted comparing the study of the maximum IUC and

MPC[1] problems in [1] to our shared-memory Algorithm 9. We used the same graph instances from DIMACS 10 clustering and partitioning challenge. Detailed results are presented in Appendix A.1. Table 3.1 shows the brief summary. The "Solved" column has the number of fully solved instances while the "Improved" column has the number of instances where better solutions were found. Our updated verifier and parallelizm show a considerable improvement of the solution quality.

Type	Total	Solved in [1]	Solved in here	Improved in here
IUC	60	25	35	29
MPC	60	27	32	21

Table 3.1: Summary of parallel RDS against [1].

Conducting a comprehensive study for the distributed-memory RDS algorithm requires a significant usage of cluster resources. Hence, we provide its performance on the maximum IUC problem for "interesting" instances, i.e., when Algorithm 9 took more than 10 seconds but less than 150 seconds to solve a problem. Table 3.2 shows the results of the computational study, with the column labels having the following meaning:

- $|V|$ - the number of vertices;
- $|E|$ - the number of edges;
- $\rho(G)$ - the edge density in percents;
- $\omega(G)$ - the clique number;
- *Problem* - the problem to solve, either the maximum unweighted IUC or MPC;
- *Time OMP* - time taken by Algorithm 9 with 15 cores on a single node;
- *Time MPI* - time taken by the distributed-memory algorithm with 4 nodes, each node has access to 15 cores and runs Algorithm 9.

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	Problem	Time OMP	Time MPI
brock200_1	200	14,834	74.54	21	IUC	31.39	16.71
hamming10-2	1,024	518,656	99.02	512	IUC	26.98	20.58
brock200_3	200	12,048	60.54	15	MPC	102.51	60.71
hamming10-2	1,024	518,656	99.02	512	MPC	64.58	39.38
keller4	171	9,435	64.91	11	MPC	10.17	3.9
san200_0.9_3	200	17,910	90.00	70	MPC	123.92	64.43
san400_0.9_1	400	71,820	90.00	100	MPC	136.85	100.52

Table 3.2: Demonstration of the distributed-memory RDS algorithm performance.

Furthermore, it is interesting to compare our approach versus the best known results for RDS algorithm so far, presented in [2]. The Algorithm 9 has the same verifiers for s-plex and s-defective clique as described in that work. The code used there is not available to the general public and it was not shared with us. Consequently, we were not able to achieve exact results as in [2]; it can be clearly seen that for graphs with considerable number of vertices their version had an advantage, most likely using implementation tricks not mentioned in the paper. Hence, we compare our results to our own serial version of the algorithm, in addition to the aforementioned work.

Moreover, there is a certain scale-reduction technique used in [2]. Suppose for a graph G we know its clique number $\omega(G)$. Then, removal of vertices with degree less than $\omega(G) - s$ does not affect maximum s-plex and $(s-1)$-defective clique. This is equivalent to finding the maximum $(\omega(G) - s)$-core of a graph and is called the *peeling procedure*, which can be done in polynomial time [80].

The detailed results are presented in Appendix A.3. The summary is briefed in Table 3.3. There are totally 239 instances studied. The columns have the following notations:

- *Solved in [2]* - the number of instances fully solved in [2] for the maximum subgraph problem;
- *Solved seq. RDS* - the number of instances fully solved by Algorithm 2;
- *Solved par. RDS* - the number of instances fully solved by Algorithm 9;

[1]Multi-partite clique: complement graph of IUC.

- *Improved [2]* - the number of instances for which Algorithm 9 finds a better solution than that reported in [2];
- *Improved seq. RDS* - the number of instances for which Algorithm 9 finds a better solution than Algorithm 2.

Problem	Solved in [2]	Solved seq. RDS	Solved par. RDS	Improved [2]	Improved seq. RDS
2-plex	172	170	177	33	25
3-plex	145	141	148	49	41
4-plex	131	110	128	55	52
5-plex	122	83	94	87	76
1-defective	181	179	182	29	19
2-defective	168	163	171	40	33
3-defective	146	142	151	43	40
4-defective	138	132	141	41	33
Total	239				

Table 3.3: Summary of parallel RDS against [2] for s-plex and s-defective clique.

Appendix A.2 provides a comprehensive report of the results for standard instances for the maximum biclique problem using Algorithm 9 and verifier from Algorithm 8. This is the first such study to the best of our knowledge. For the real-life instances, a scale reduction described in [81] can be applied to efficiently reduce input graph sizes.

4. COMPUTING HADWIGER'S NUMBER

4.1 Background and research questions

In this chapter we investigate questions about computing Hadwiger's number of a graph. An edge contraction is an operation when two endpoints of an edge are replaced with a single vertex which is adjacent to their neighbors, removing multiedges if necessary. A new graph obtained by series of edge contractions from a graph G is called a minor of G. Given a simple, undirected, and connected graph G, its *Hadwiger's number* $h(G)$ is the size of the largest clique that can be found in a minor of G. The *chromatic number* $\chi(G)$ of a graph G is the minimum number of colors needed to color its vertices such that any two adjacent vertices have different colors. Back in 1943, Swiss mathematician Hugo Hadwiger formulated one of the most withstanding conjectures in graph theory, stating that for every graph G it holds that $\chi(G) \leq h(G)$.

The case of $h(G) \leq 4$ was proved by Hadwiger himself while stating the conjecture. The famous four-color theorem[1] [82] implies that the case $h(G) = 5$ indeed holds [83]. Much time later, the solution for $h(G) = 6$ was presented, interestingly, again involving the four-color theorem [84]. It remains unsolved for all other values of $h(G)$ for a long time, and is even called "one of the deepest unsolved problems in graph theory" by Bollobás and Erdős [85]. An excellent survey of recent developments can be found in [86].

The problem of finding Hadwiger's number of a graph was not extensively studied. From general graph minor theory, it follows that it is fixed-parameter tractable [87]. Recently, it was shown to be $\mathcal{NP}$-hard [88].

In the case when a graph is disconnected, we define Hadwiger's number to be the largest Hadwgier's number of connected components.

[1] Four color theorem states that every planar graph has the chromatic number at most 4.

4.1.1 Separators

For integer programming formulations, discussed in this chapter, we need the concept of an a, b-separator.

Definition 20. *A subset $C \subseteq V \setminus \{a, b\}$ of vertices is called an a, b-separator for $G = (V, E)$ if there is no a, b-path in $G[V \setminus C]$.*

An a, b-separator is *minimal*, if there is no other a, b-separator which is its strict subset. Given a connected subgraph and a vertex not in it, a minimal separator could be found in linear time [89] with Algorithm 11.

Algorithm 11 Find a minimal separator between a connected subgraph and a vertex.

1: Given $C_i \subset V$, such that $G[C_i]$ is connected; $j \notin C_i$
2: Compute $A(C_i) = \{v \in V \mid \exists w \in C_i \ \{v, w\} \in E\}$
3: Find the set R_j of nodes reachable from j in $G[V \setminus C_i]$
4: **return** $R_j \cap A(C_i)$

Finally, define the set $\mathbb{SEP}$ of all minimal separators. Notice, that there might be an exponential number of them, hence it is intractable to generate all separators.

$$\mathbb{SEP} := \left\{ (a, b, C) \;\middle|\; \{a, b\} \in \binom{V}{2} \setminus E;\ a < b;\ C \text{ is an } a, b\text{-separator} \right\}. \tag{4.1}$$

4.1.2 Basic observations

There is an alternative definition of Hadwiger's number $h(G)$ of a graph. It can be defined as the largest integer k for which there exists a partition $V_1, \ldots, V_k$ of the set of vertices V such that:

1. for each $i = 1, \ldots, k$, the subgraph $G[V_i]$ is connected, and
2. for every $\{i, j\} \in \binom{[k]}{2}$, there exists an edge from E with one endpoint in V_i and the other endpoint in V_j.

We call the first condition *contiguity*, i.e., every partition must be connected or contiguous; the second one is *proximity*, i.e., between every two partitions there must exist an edge.

The equivalence between clique minor definition and partitioning is easy to see. Given a partitioning, because of contiguity it is possible to contract every partition into a single vertex, and proximity guarantees that the resulting graph is a clique. In order to obtain partitioning from contractions, reverse the contraction process for the clique minor, mark edges instead of contracting, and obtain contiguous partitions.

Theorem 15. *Given a graph $G = (V, E)$, for its Hadwiger's number $h(G)$ it holds that*

$$h(G) \leq \left\lfloor \frac{3 + \sqrt{9 + 8(|E| - |V|)}}{2} \right\rfloor. \tag{4.2}$$

Proof. Suppose an optimal partitioning of G has partitions $V_1, V_2, \ldots, V_k$ where $k = h(G)$. There are at least $\binom{k}{2}$ edges between partitions and each partition has at least $|V_i|-1$ edges for $i = 1, \ldots, k$ to be connected. Hence,

$$\binom{k}{2} + \sum_{i=1}^{k} (|V_i| - 1) \leq |E|.$$

Solving this inequality and taking into account that $\sum_{i=1}^{k} |V_i| = |V|$ completes the proof. □

Observation 6. *Given a graph $G = (V, E)$, for its Hadwiger's number $h(G)$ it holds that*

$$h(G) \geq \omega(G).$$

Proof. Contract all edges except those between vertices in a maximum clique. □

In addition, we notice that viewing Hadwiger's number as a partition suggests an interesting exact algorithm. First of all, we call a partition feasible if it satisfies contiguity and proximity. Suppose we are given an optimal partition. Merging any two parts preserves feasibility, and hence the solution can be found by iteratively splitting any feasible partition, for example, the trivial one with a single component which equals to the entire graph. Secondly, iterating over all partitions

in lexicographical order makes pruning efficient, because if a single part is not contiguous or proximity is not satisfied, the whole branch can be pruned. For the sake of brevity we do not present the entire algorithm based on these ideas, as it has demonstrated inferior performance compared to certain integer programming models. Details can be found in Appendix B.3.

4.2 Integer formulations

4.2.1 Base formulation

For each partition V_i, we can select a single node $r \in V_i$ to be its representative or clusterhead. We use binary variables x_{ij} for all $i, j \in V$. To break the symmetry in the model, we say that the clusterhead must be the node with the highest index in that partition. Hence, by our conventions, x_{ij} is only defined when $i \leq j$. Setting $x_{ij} = 1$ means that node i has selected node j to be the representative for its partition. Setting $x_{ii} = 1$ means i is a clusterhead and the highest-index node in its partition, and $\sum_{i \in V} x_{ii}$ is the number of partitions.

The $BASE$ model is defined as follows.

$$\max \sum_{i \in V} x_{ii} \tag{4.3a}$$

$$x_{ij} \leq x_{jj} \qquad i, j \in V : \ i < j \tag{4.3b}$$

$$\sum_{j=i}^{n} x_{ij} = 1 \qquad i \in V \tag{4.3c}$$

$$x \text{ satisfies contiguity} \tag{4.3d}$$

$$x \text{ satisfies proximity} \tag{4.3e}$$

$$x_{ij} \in \{0, 1\} \qquad i, j \in V : \ i < j. \tag{4.3f}$$

Constraints (4.3b) ensure that node i can be assigned to node j only if j is selected as a clusterhead, and constraints (4.3c) ensure that each node i is assigned to some clusterhead. The summation starts at $j = i$ to impose that i is not assigned to an earlier node.

Next subsections present different ways to impose contiguity and proximity. The MCF type

models use multi-commodity flow type formulations and CUT type models use a, b-separators [90].

4.2.2 Contiguity

Contiguity models are extensively studied in many districting application papers, especially in [91]. Here we present a brief survey of those models. We use the notation $\delta^-(v)$ for total flow into node v and $\delta^+(v)$ for total flow out of node v, i.e., for some f it holds that $f(\delta^-(v)) = \sum_{\{i,v\}\in E} f(i,v)$ and $f(\delta^+(v)) = \sum_{\{v,i\}\in E} f(v,i)$.

The MCF0 model comes from [92], where a flow-based formulation adapted from [93] is proposed. The idea is that every clusterhead emits flow of its type, and if a node is assigned to it, then some flow must be consumed. Define the flow variables for every $v \in V$ and for $\{i,j\} \in E$ when $i \leq v, j \leq v$.

$$f_{ij}^v = \text{the amount of flow, originating at clusterhead } v, \text{ that is sent across edge } \{i,j\}.$$

The model is described as following:

$$x \in BASE \tag{4.4a}$$

$$f^j(\delta^-(i)) - f^j(\delta^+(i)) = x_{ij} \qquad i < j,\ j \in V \tag{4.4b}$$

$$f^j(\delta^-(i)) \leq (|V|-1)x_{ij} \qquad i < j,\ j \in V \tag{4.4c}$$

$$f^j(\delta^-(j)) = 0 \qquad j \in V \tag{4.4d}$$

$$f_{ij}^v \geq 0 \qquad \{i,j\} \in E,\ i \leq v,\ j \leq v,\ v \in V. \tag{4.4e}$$

Constraints (4.4b) tell that every node in a clusterhead must consume 1 unit of flow of its clusterhead type, constraints (4.4c) limit the in-flow, constraints (4.4d) prohibit clusterhead to consume any flow.

Authors in [91] noticed that constraints (4.4c) are a big-M type and might not be efficient for

IP solvers. Hence, they can be tweaked making certain flow binary, such as

$$0 \leq f_{ij}^{v} \leq f_{ij}^{j} \qquad \forall v \leq j,\ \forall \{i,j\} \in E$$
$$f_{ij}^{j} \in \{0,1\} \qquad \forall \{i,j\} \in E.$$

Another important and tighter than MCF0 model is called MCF [91]. We extend the flow type to be sent from a clusterhead to a specific vertex. Define the flow variables for $b \in V$, $a < b$, and every edge $\{i,j\} \in E$ with $i \leq b, j \leq b$.

$$f_{ij}^{ab} = \begin{cases} 1 & \text{if edge } \{i,j\} \in E \text{ is on the path to vertex } a \text{ from its clusterhead } b \\ 0 & \text{otherwise.} \end{cases}$$

The flow variables are not binary, but there is an optimal solution making them integral. The MCF model is formulated as follows.

$$x \in BASE \tag{4.5a}$$
$$f^{ab}(\delta^{+}(b)) - f^{ab}(\delta^{-}(b)) = x_{ab} \qquad \forall a < b,\ \forall b \in V \tag{4.5b}$$
$$f^{ab}(\delta^{+}(i)) - f^{ab}(\delta^{-}(i)) = 0 \qquad i < b,\ i \neq a, a < b,\ \forall b \in V \tag{4.5c}$$
$$f^{ab}(\delta^{-}(b)) = 0 \qquad a < b,\ \forall b \in V \tag{4.5d}$$
$$f^{ab}(\delta^{-}(j)) \leq x_{jb} \qquad j < b,\ a < b,\ \forall b \in V \tag{4.5e}$$
$$f_{ij}^{ab} \geq 0 \qquad \{i,j\} \in E,\ i \leq b,\ j \leq b,\ a < b,\ b \in V. \tag{4.5f}$$

The CUT formulation uses a, b-separators for the constraints. Even though it is intractable to generate the entire model because of an exponential number of its constraints, the so-called "lazy" approach can be used. Whenever a solver finds an integral solution, if it is not feasible, Algorithm 11 can be used to generate a constraint for separation. Constraints (4.6b) say that if a is

in partition of b, then there exists a path from b to a.

$$x \in BASE \tag{4.6a}$$

$$x_{ab} \leq \sum_{c \in C} x_{cb} \qquad (a, b, C) \in \mathbb{SEP}. \tag{4.6b}$$

Separators-based models are shown to be very tight [90, 89]. The nice relationship between models was shown in [91] and formulated in Theorem 16.

Theorem 16. $CUT = proj_x MCF \subseteq proj_x MCF0$.

The original proof is for formulations that do not break the model symmetry. We present a similar proof in Theorem 17 for our formulations.

4.2.3 Proximity

The straightforward way of enforcing the proximity is by its definition, i.e., between every two partitions there must exist an edge. This makes the following EDGE model.

$$x \in BASE \tag{4.7a}$$

$$\sum_{(u,v) \in E} x_{ui} x_{vj} \geq x_{ii} + x_{jj} - 1 \qquad i < j, i \in V, j \in V \tag{4.7b}$$

The constraint (4.7b) is not linear and needs to be linearized. We introduce new binary variables $y_{uivj} = x_{ui} x_{vj}$. The resulting model is referred to as EDGE_LINEAR.

$$\sum_{(u,v) \in E, u \leq i, v \leq j} y_{uivj} \geq x_{ii} + x_{jj} - 1 \qquad \forall i < j, i \in V, j \in V \tag{4.8a}$$

$$y_{uivj} \leq x_{ui} \qquad \forall u < i, v < j \tag{4.8b}$$

$$y_{uivj} \leq x_{vj} \qquad \forall u < i, v < j \tag{4.8c}$$

$$y_{uivj} \in \{0, 1\} \tag{4.8d}$$

Our preliminary experiments showed that EDGE_LINEAR is not a tight model which performs

poorly. Hence, we present a flow-based and a cut-based formulations similar to MCF and CUT. The main idea is based on observation that saying that two partitions have an edge in between and are contiguous is the same as saying that the partition formed by their union is contiguous.

$$f_{ij}^{ab} = \begin{cases} 1 & \text{if edge } \{i,j\} \in E \text{ is on the path to clusterhead } a \text{ from clusterhead } b \\ 0 & \text{otherwise.} \end{cases}$$

The idea is similar to MCF. For every two clusterheads $a < b$, the larger one (b) emits flow and the smaller one (a) must consume it; the flow can go only through vertices in either a or b partition. The corresponding formulation is called PMCF:

$$x \in BASE \tag{4.9a}$$

$$f^{ab}(\delta^+(b)) - f^{ab}(\delta^-(b)) \geq x_{aa} + x_{bb} - 1 \qquad a \in V \setminus N[b],\ a < b,\ b \in V \tag{4.9b}$$

$$f^{ab}(\delta^+(i)) - f^{ab}(\delta^-(i)) = 0 \qquad i < b,\ i \neq a,\ a \in V \setminus N[b],\ a < b,\ b \in V \tag{4.9c}$$

$$f^{ab}(\delta^-(b)) = 0 \qquad a \in V \setminus N[b],\ a < b,\ b \in V \tag{4.9d}$$

$$f^{ab}(\delta^-(j)) \leq 1_{\{j<a\}} x_{ja} + x_{jb} \qquad j \in V \setminus \{b\},\ a \in V \setminus N[b],\ a < b,\ b \in V \tag{4.9e}$$

$$f_{ij}^{ab} \geq 0 \qquad \{i,j\} \in E,\ a \in V \setminus N[b],\ a < b,\ b \in V. \tag{4.9f}$$

Here $N[v]$ for $v \in V$ is the *closed neighborhood of* v, defined as $N[v] = N(v) \cup \{v\}$. Constraints (4.9b) tell to emit flow type ab only when a and b are clusterheads. Constraints (4.9c) are flow preserving. Constraints (4.9d) prohibit clusterhead b to consume flow. Constraints (4.9e) allow flow to go only through vertices assigned to either a or b partitions.

The separator-based version is similar to CUT. The idea is to enforce every two clusterheads a and b to be connected with a path which consists of vertices assigned to one of corresponding

parts (4.10b). We formulate PCUT as follows:

$$x \in BASE \tag{4.10a}$$

$$\sum_{c \in C: c<a} x_{ca} + \sum_{c \in C: c<b} x_{cb} \geq x_{aa} + x_{bb} - 1 \qquad (a, b, C) \in \mathbb{SEP}. \tag{4.10b}$$

Using similar ideas as for contiguity, PMCF is the extended formulation [94] of PCUT.

Theorem 17. $PCUT = proj_x PMCF$.

Proof. The proof is a modified version of a similar theorem in [91].

($\supseteq$) Suppose that $(\hat{x}, \hat{f})$ belongs to PMCF. Let us show that $\hat{x}$ belongs to (4.10b). For arbitrary non-adjacent vertices a and b, $a < b$, a, b-separator $C \subseteq V \setminus \{a, b\}$, and set B of vertices reachable from b in $G[V \setminus C]$, we have:

$$x_{aa} + x_{bb} - 1 \leq f^{ab}(\delta^+(b)) - f^{ab}(\delta^-(b)) \tag{4.11a}$$

$$\leq f^{ab}(\delta^+(B)) - f^{ab}(\delta^-(B)) \tag{4.11b}$$

$$\leq f^{ab}(\delta^+(B)) \tag{4.11c}$$

$$\leq f^{ab}(\delta^-(C)) \tag{4.11d}$$

$$\leq \sum_{j \in C} 1_{\{j<a\}} x_{ja} + x_{jb}. \tag{4.11e}$$

Inequality (4.11b) holds because of (4.9c), (4.11c) holds because f are non-negative, and (4.11d) holds because the flow through C is the flow emitted from b.

($\subseteq$) Suppose that $\hat{x}$ satisfies (4.10b). For every pair of distinct non-adjacent vertices $a, b \in V$,

we define $\hat{f}^{ab}$ as the solution to the following flow problem.

$$\begin{aligned}
&\max f^{ab}(\delta^+(b)) \\
&f^{ab}(\delta^+(i)) - f^{ab}(\delta^-(i)) = 0 && \forall i \in V \setminus \{a, b\} \\
&f^{ab}(\delta^-(b)) = 0 \\
&f^{ab}(\delta^-(j)) \leq 1_{j<a}\hat{x}_{ja} + \hat{x}_{jb} && \forall j \in V \setminus \{b\} \\
&f^{ab}_{ij} \geq 0 && \forall (i, j) \in A.
\end{aligned}$$

This is a flow problem from b with vertex capacities $1_{j<a}\hat{x}_{ja} + \hat{x}_{jb}$. By a classical result [95, Section 1.11], the optimal value equals to the minimum weight a, b-cut C, i.e.,

$$\hat{f}^{ab}(\delta^+(b)) = \sum_{i \in C} 1_{j<a}\hat{x}_{ja} + \hat{x}_{jb},$$

which is greater than $x_{aa} + x_{bb} - 1$ by (4.10b). Hence, $\hat{f}^{ab}$ satisfies all constraints for PMCF. □

4.2.4 The chromatic number

The idea of finding the chromatic number of a graph $\chi(G)$ using partitions was presented in [96]. By slightly modifying the model, one can come up with an integer program which looks like our Hadwiger's number formulations. Hence, Hadwiger's conjecture can be formulated in terms of objectives for similar integer programs.

The alternative definition of chromatic number $\chi(G)$ is the following. It can be defined as the smaller integer k for which there exists a partition $V_1, \ldots, V_k$ of the vertices V such that for each two vertices $v \in V_i, w \in V_i$ for every i it holds that $\{v, w\} \notin E$. Moreover, the optimal solution satisfies the proximity requirement as explained in Observation 7.

Observation 7. *The optimal partition for chromatic number problem satisfies the proximity requirement.*

Proof. Suppose there exists an optimal partition $V_1, \ldots, V_k$ and there is no edge between V_1 and

V_2. Consider partition $V_1 \cup V_2, V_3, \ldots, V_k$. No two nodes in $V_1 \cup V_2$ are adjacent; if they are from the same part, by setup they are not adjacent; if they are from different parts, there must be no edge by assumption. Hence, this is a valid chromatic partition which has one less part as the optimal, which is a contradiction. □

Hence, an integer program for finding the chromatic number $\chi(G)$ can be formulated as follows, with the proximity requirement being redundant.

$$\min \sum_{i \in V} x_{ii} \tag{4.12a}$$

$$x_{ij} \leq x_{jj} \qquad i, j \in V,\ i < j \tag{4.12b}$$

$$\sum_{j=i}^{n} x_{ij} = 1 \qquad i \in V \tag{4.12c}$$

$$x_{ik} + x_{jk} \leq 1 \qquad i < k,\ j < k,\ \{i, j\} \in E,\ k \in V \tag{4.12d}$$

$$x \text{ satisfies proximity.} \tag{4.12e}$$

Constraints (4.12b)-(4.12c) specify a partition, constraints (4.12d) make sure that no edge belongs to a single partition, constraints (4.12e) are redundant due to Observation 7, objective (4.12a) minimizes the number of parts.

Conclusively, it might be fruitful to consider partitioning with proximity only in the future work.

4.3 Scale reduction

Oftentimes, real-life networks are sparse and have a so-called "small-world" property [27]. In this case it is fruitful to look for fast "preprocessing" algorithms to significantly reduce the instance size [10]. For Hadwiger's number, one of the possible reductions is based on the biconnected components.

Definition 21. *A biconnected component in graph G is the maximal biconnected subgraph in G. A connected graph is biconnected if removal of any single vertex keeps it connected.*

Any connected graph decomposes into a tree of biconnected components that are attached to each other at shared vertices called *articulation vertices*.

The classical algorithm to find all biconnected components of a graph is presented in [97]. It is based on depth-first search and thus runs in linear time.

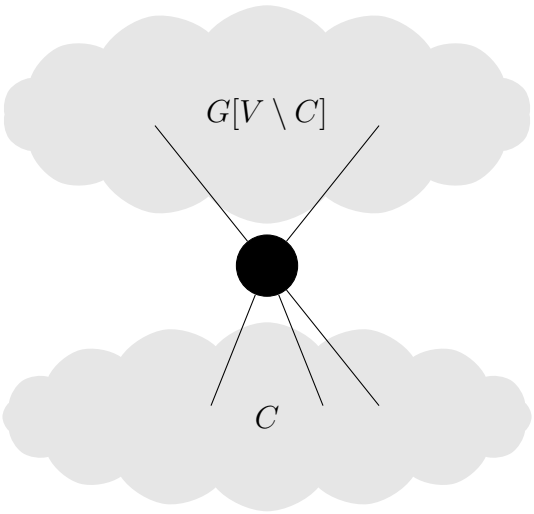


Figure 4.1: Biconnected components with articulation vertex belonging to both

Theorem 18. *Let graph G have biconnected components $C_1, C_2, \ldots, C_k$. Then*

$$h(G) = \max\{h(G[C_1]), h(G[C_2]), \ldots, h(G[C_k])\}.$$

Proof. For a simple, undirected graph $G = (V, E)$ consider a biconnected component C_1 and the remaining graph $G[V \setminus C_1]$. Notice that contracting edges will not introduce a new edge between them. Hence, the maximum clique minor must have either $G[C]$ or $G[V \setminus C]$ entirely contracted, so $h(G) = \max\{h(G[C], h(G[V \setminus C]\}$. Repeat the argument for $G[V \setminus C]$. □

Corollary 3. *Vertices of degree 1 can be removed from a graph G with more than 2 vertices without affecting $h(G)$.*

Observation 8. *Vertices of degree 2 can be eliminated from a graph G with $h(G) > 3$, by contracting one of the edges it is incident to, without affecting $h(G)$.*

Proof. If $h(G) > 3$, one such vertex cannot be included in a cluster by itself, hence at least one of the two edges must be contracted. Contracting any of the two edges leads to the same resulting graph. □

Results in Table B.1 suggest that many real life instances have a single giant biconnected component. We also report the size of the biconnected 3-core obtained from the largest biconnected 2-core by eliminating vertices of degree 2 using the contraction of one of its incident edges. This figure gives the number of vertices in the largest among the biconnected graphs we have to analyze for the given instance.

4.4 Heuristic

After contracting an edge $\{v_1, v_2\} \in E$, the edge density changes from $\frac{|E|}{\binom{|V|}{2}}$ to $\frac{|E|-1-|N(v_1)\cap N(v_2)|}{\binom{|V|-1}{2}}$. It is easy to see that the maximum increase of the density is achieved by contracting an edge $\{v_1, v_2\}$ for which $|N(v_1) \cap N(v_2)|$ is minimal. Thus, a natural greedy approach tries to contract the edge with minimum $|N(v_1) \cap N(v_2)|$, as we aim to increase the edge density as much as possible.

Algorithm 12 A heuristic for finding the Hadwiger's number.

```
1: ĥ(G) ← |V|
2: while G is not a complete graph do
3:     find the edge {u, v} ∈ E with minimum |N(u) ∩ N(v)|
4:     G ← G/uv                                  ▷ Contract {u, v}
5:     ĥ(G) ← ĥ(G) − 1
6: return ĥ(G)
```

Contracting an edge in the graph changes the common neighborhood locally. Indeed, if an edge $\{a, b\}$ is contracted, only the edges whose endpoint has either a or b in its neighborhood

might change the common neighborhood metric. This suggests recomputing the intersection size only for a small number of edges.

A natural question is, whether the proposed heuristic provides any guarantee on the quality of the computed solution $\hat{h}(G)$. Unfortunately, the following negative results were discovered by a brute force check on all graphs (non-isomorphic, connected) with up to 10 vertices.

- The output $\hat{h}(G)$ can be less than the trivial bound given by the clique number $\omega(G)$ (and hence, less than the chromatic number $\chi(G)$).
- There exists a graph for which an edge with non-maximum common neighborhood must be contracted, while none of edges with maximum common neighborhoods can be contracted to achieve optimality (e.g., just focusing on tie-breaking rules will not make Algorithm 12 optimal).

4.5 Lagrangian relaxation

One of the powerful classical methods for finding estimates of the optimal value is based on Lagrangian relaxation [23]. It is known that Lagrangian relaxation provides bounds that are at least as tight as those obtained using the linear relaxation [98]. We will show that the inner problem for Hadwiger's number is $\mathcal{NP}$-hard and, consecutively, this approach is intractable.

For simplicity of the argument, we will consider BASE model without breaking the symmetry, i.e., a clusterhead is not necessarily the highest-index node in a partition. Moving constraints (4.3c) to the objective using Lagrangian multipliers $u_i,\ i \in V$, and grouping coefficients, the problem becomes

$$\sum_{i=1}^{|V|} u_i + \max_x \sum_{i=1}^{|V|} \left(x_{ii} - \sum_{j=1}^{|V|} u_j x_{ji} \right) \tag{4.13a}$$

$$x \text{ satisfies contiguity} \tag{4.13b}$$

$$x \text{ satisfies proximity} \tag{4.13c}$$

$$x_{ij} \in \{0, 1\} \qquad i, j \in V. \tag{4.13d}$$

The problem splits into subproblems for different i. Each of the subproblems can be formulated as follows: given node weights $w_j = 1_{\{j=i\}} - u_j$, find the maximum weight connected subgraph. This problem is known to be $\mathcal{NP}$-hard even on planar graphs with weights ± 1 [99].

4.6 Computational results

We tried to find the exact solutions for instances from standard graph coloring and clustering instances. Our computer is equipped with Intel® Xeon® CPU at 2.40GHz, 8 GB RAM. The code is compiled with GNU C++ compiler version 4.9.2. The IP solver is GUROBI version 8.1.

We were able to solve only 8 instances. The IP model producing the obtained results is the CUT-PCUT one with so-called "lazy"-type constraints; we found that the best strategy is to add one cut at a time. Given an integer solution by GUROBI, our callback checks the feasibility of partition. If contiguity is violated, we add a cut and let IP solver proceed; if proximity is violated, we also add the corresponding cut. Callbacks take little to no time to separate a solution; branching through integer points appears to be the bottleneck though.

Table 4.1 reports the results. The columns in this table are denoted as follows:

- $|V|$ - the number of vertices;
- $|E|$ - the number of edges;
- $\rho(G)$ - the edge density;
- $\chi(G)$ - the chromatic number;
- $h(G)$ - Hadwiger's number;
- *Time* - the number of seconds GUROBI took to find the optimum solution;
- *Callbacks* - the number of callbacks calls;
- *Callback time* - total cumulative time spent on callbacks.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$h(G)$	Time	Callbacks	Callback Time
1-FullIns_3	30	100	0.23	4	11	61,212	14,503	1.26
myciel3	11	20	0.36	4	6	0.05	80	0.00
myciel4	23	71	0.28	5	10	53.06	1,652	0.08
queen5_5	25	160	0.53	5	14	11.45	841	0.05
r125.1c	125	7,501	0.97	46	85	37.87	296	0.06
MANN_a9	45	918	0.93	18	30	4.09	86	0.01
karate	34	78	0.14	5	6	7.13	3,116	0.1
johnson-8-2-4	28	210	0.54	6	16	31.10	463	0.03

Table 4.1: Hadwiger's number for standard benchmark instances.

5. SUMMARY, CONCLUSIONS, AND FUTURE WORK

This dissertation is centered around cluster detection and partitioning problems in networks. Below, we revisit our contributions and provide some ideas about future research directions.

In Chapter 2 we studied the behavior of the maximum IUC problem and its generalization, the maximum IUΠ problem, on uniform random graphs. We showed that the former problem has a logarithmic upper bound, similarly to the maximum clique problem, which implies a subexponential algorithm. Moreover, stronger results were presented: for dense graphs, a solution to the maximum IUC problem consists of a maximum clique and, maybe, some additional comparatively "small" clusters. When randomness is balanced, i.e., $p = 0.5$, the probability of a graph to form an IUC is can be described with an expression looking similar to that for the maximum clique, but also involving the n-th Bell number, where n is the number of nodes in the graph. For the maximum IUΠ problem, we present a set of sufficient conditions to check if for a given property Π our logarithmic bound holds. We applied this method for a set of different properties such as s-plex, s-defective clique, and γ-quasi clique. In the end of the chapter, we presented an alternative approach for finding upper bounds, utilizing the result for the maximum IUC problem. The additional result covers the case when clusters are not completely independent, i.e., when there still might be a fixed number of edges between them.

One of the interesting related research tasks is to develop an algorithm for finding the maximum IUC on uniform random graphs in a greedy manner. Some of the algorithms for finding large cliques are known [100], however there is still a gap between known bounds.

Additional research might focus on power-law graphs, which appear more often in practice. We conjecture that behavior of an IUC will have nicely described properties, as those graphs have definite clustering characteristics. Also, algorithms which run especially well on this type of networks are immensely valuable in practical applications.

Another research direction concerning the maximum IUC problem consists in developing efficient scale reduction techniques. The following idea could be tried. It is known, that a large, sparse

instance of the maximum clique problem can often be reduced to a much smaller instance using degree and neighborhood related metrics [101]. The IUC problem is a so-called weak independent set problem on the 3-uniform hypergraphs, where a single hyperedge corresponds to an open triangle in the original graph. Evaluating "degrees", "neighborhoods", and other novel structures in 3-uniform hypergraphs may lead to scale reductions sought.

In Chapter 3 we revisited the Russian Doll Search algorithm. We proposed two directions to parallelize it while preserving its signature pruning steps: parallelizing the first branching level or parallelizing its main loop. We used shared-memory and distributed-memory models to implement these ideas. Using a parallel version of the algorithm makes the comparison to IP-based approaches more fair, as typically IP solvers utilize parallel capabilities of modern computers. In addition, we emphasized an important property of a verifier that may drastically improve the running time. Based on this observation, we improved the verifier for the maximum IUC problem from quadratic time to linear, and came up with a constant time verifier for the maximum biclique problem.

The very first question left unanswered concerns the efficiency of our parallelizations. We do not claim that our approaches keep CPU utilization at highest levels and not idling. Hence, any insights in this direction might be interesting for enhancing parallel search.

Another direction to explore is motivated by Chapter 2. Given a verifier for any property Π, can we efficiently build a new verifier for IUΠ? For a clique we know that there is a linear time one. However, checking every connected (or, even worse, independent) component to satisfy property Π appears to be inefficient.

In Chapter 4 we presented the first computational study for the problem of finding the Hadwiger's number of a graph. A closely related conjecture is well-know to graph theorists and mathematicians, but attracted limited attention in Operations Research community. We split the problem into two subptasks: enforcing contiguity and proximity. We presented several IP formulations and a projection relation between them. The problem remains extremely hard even on small graphs, and a state of the art IP solver was able to solve only small instances with advanced usage of lazy-type constraints. In addition, we presented a basic scale-reduction algorithm based

on biconnected components of a graph, as well as a quick greedy heuristic. We provided results of extensive computational experiments with a testbed consisting of standard graph clustering and coloring instances (see Appendix B).

There are still many unanswered questions regarding Hadwiger's number of a graph, putting aside the famous conjecture. The first one concerns an effective exact combinatorial algorithm. The main challenge is that the number of different partitions is much larger than the number of subsets, making efficient pruning strategies paramount. We proposed a scheme for such algorithms in Section B.3. Another important direction of future research consists in developing more efficient scale reduction techniques. It is natural to expect that finding Hadwiger's number is an "easy" problem on sparse graphs, hence suitable degree/degeneracy techniques may exist. We propose an interesting idea to reduce the size of a network for computing its Hadwiger's number. Suppose we have a good upper bound on the solution. Then, setting x_{ii} to either 0 or 1 and comparing integer relaxation to that bound, we might be able to fix its value permanently. Hence, it could be useful to be able to compute such bounds. One of the methods to try is semi-definite programming, which can potentially utilize our EDGE formulation. In addition, it would be interesting to investigate the relationship between EDGE and PCUT formulations. Results of our experiments suggest that PCUT is stronger than EDGE.

Even though Hadwiger's number is important from a theoretical perspective, we believe that it should have various real-life applications because of the correspondence to a certain optimal partitioning. Here we provide a potential application scenario. Suppose we are given a partitioning which satisfies contiguity and proximity. Then every two nodes are: i) in the same partition and there exists a path between them; ii) in different partitions and there exists a path with at most one edge which does not belong to a single partition. We call edges that fully belong to a partition and are necessary for contiguity (partition's tree) "secured", and all other edges "unsecured". Then, partitioning corresponding to Hadwiger's number ensures that there is a path with at most one "unsecured" edge between every two nodes. Consider a network to be a communication network with the ability to secure edges. Given a completely unsecured network, what is the minimum

number of edges we have to make secure, so that the network becomes “almost secured”? This question demonstrates a strong connection to the problem of finding Hadwiger’s number.

REFERENCES

[1] Z. Ertem, E. Lykhovyd, Y. Wang, and S. Butenko, "The maximum independent union of cliques problem: complexity and exact approaches," *Journal of Global Optimization*, pp. 1–18, 2018.

[2] T. Gschwind, S. Irnich, and I. Podlinski, "Maximum weight relaxed cliques and Russian Doll Search revisited," *Discrete Applied Mathematics*, 2016.

[3] L. Euler, "Solutio problematis ad geometriam situs pertinentis," *Commentarii academiae scientiarum Petropolitanae*, pp. 128–140, 1741.

[4] V. Boginski, S. Butenko, and P. M. Pardalos, "Mining market data: a network approach," *Computers & Operations Research*, vol. 33, no. 11, pp. 3171–3184, 2006.

[5] P. Uetz, T. Ideker, and B. Schwikowski, "Visualization and integration of protein-protein interactions," *Protein-Protein Interactions–a Molecular Cloning Manual*, 2002.

[6] Y. Iwasaki, A. G. Kusne, and I. Takeuchi, "Comparison of dissimilarity measures for cluster analysis of x-ray diffraction data from combinatorial libraries," *npj Computational Materials*, vol. 3, no. 1, p. 4, 2017.

[7] R. Diestel, *Graph Theory*, vol. 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, 5 ed., 2017.

[8] J. Pattillo, N. Youssef, and S. Butenko, "On clique relaxation models in network analysis," *European Journal of Operational Research*, vol. 226, no. 1, pp. 9–18, 2013.

[9] Z. Ertem, A. Veremyev, and S. Butenko, "Detecting large cohesive subgroups with high clustering coefficients in social networks," *Social Networks*, vol. 46, pp. 1–10, 2016.

[10] A. Verma, A. Buchanan, and S. Butenko, "Solving the maximum clique and vertex coloring problems on very large sparse networks," *INFORMS Journal on Computing*, vol. 27, no. 1, pp. 164–177, 2015.

[11] A. Verma and S. Butenko, "Network clustering via clique relaxations: A community based approach.," *Graph Partitioning and Graph Clustering*, vol. 588, p. 129, 2012.

[12] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, "Column generation algorithms for exact modularity maximization in networks," *Phys. Rev. E*, vol. 82, p. 046112, Oct 2010.

[13] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Mathematical Programming*, vol. 79, pp. 191–215, Oct 1997.

[14] J. Chen and J. Meng, "A 2k kernel for the cluster editing problem," *Journal of Computer and System Sciences*, vol. 78, no. 1, pp. 211–220, 2012.

[15] F. Hüffner, C. Komusiewicz, H. Moser, and R. Niedermeier, "Fixed-parameter algorithms for cluster vertex deletion," *Theory of Computing Systems*, vol. 47, no. 1, pp. 196–217, 2010.

[16] A. Boral, M. Cygan, T. Kociumaka, and M. Pilipczuk, "A fast branching algorithm for cluster vertex deletion," *Theory of Computing Systems*, vol. 58, no. 2, pp. 357–376, 2016.

[17] Euclid, *Elements*. Approx. 300 B.C.

[18] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*, vol. 24. Springer Science & Business Media, 2003.

[19] L. G. Khachiyan, "A polynomial algorithm in linear programming," in *Doklady Academii Nauk SSSR*, vol. 244, pp. 1093–1096, 1979.

[20] N. Shor, "New development trends in nondifferentiable optimization," *Cybernetics and Systems Analysis*, vol. 13, no. 6, pp. 881–886, 1977.

[21] G. B. Dantzig, "Origins of the simplex method," tech. rep., STANFORD UNIV CA SYSTEMS OPTIMIZATION LAB, 1987.

[22] L. Wolsey, *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization, Wiley, 1998.

[23] L. A. Wolsey and G. L. Nemhauser, *Integer and combinatorial optimization*. John Wiley & Sons, 2014.

[24] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer programming*, vol. 271. Springer, 2014.

[25] D. W. Matula, *The largest clique size in a random graph*. Department of Computer Science, Southern Methodist University, 1976.

[26] B. Bollobás and P. Erdös, "Cliques in random graphs," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 80, pp. 419–427, Cambridge Univ. Press, 1976.

[27] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, p. 440, 1998.

[28] B. Bollobás, "Random graphs," in *Modern Graph Theory*, pp. 215–252, Springer, 1998.

[29] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci.*, vol. 5, no. 1, pp. 17–60, 1960.

[30] B. Bollobás and B. Béla, *Random graphs*. No. 73, Cambridge university press, 2001.

[31] P. Balister, B. Bollobás, J. Sahasrabudhe, and A. Veremyev, "Dense subgraphs in random graphs," *Discrete Applied Mathematics*, 2019.

[32] A. Veremyev, V. Boginski, P. A. Krokhmal, and D. E. Jeffcoat, "Dense percolation in large-scale mean-field random networks is provably "explosive"," *PloS one*, vol. 7, no. 12, pp. 1–14, 2012.

[33] M. Yannakakis, "Node-and edge-deletion $\mathcal{NP}$-complete problems," in *Proceedings of the tenth annual ACM symposium on Theory of computing*, pp. 253–264, ACM, 1978.

[34] A. Pyatkin, E. Lykhovyd, and S. Butenko, "The maximum number of induced open triangles in graphs of a given order," *Optimization Letters*, Oct 2018.

[35] A. W. Goodman, "On sets of acquaintances and strangers at any party," *The American Mathematical Monthly*, vol. 66, no. 9, pp. 778–783, 1959.

[36] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[37] T. Schank and D. Wagner, "Finding, counting and listing all triangles in large graphs, an experimental study," in *International workshop on experimental and efficient algorithms*, pp. 606–609, Springer, 2005.

[38] F. Chung and R. Graham, *Erdös on Graphs: his legacy of unsolved problems*. AK Peters/CRC Press, 1998.

[39] P. Erdös and G. Szekeres, "A combinatorial problem in geometry," *Compositio mathematica*, vol. 2, pp. 463–470, 1935.

[40] D. Kleitman and L. Pachter, "Finding convex sets among points in the plane," *Discrete & Computational Geometry*, vol. 19, no. 3, pp. 405–410, 1998.

[41] G. Tóth and P. Valtr, *The Erdös-Szekeres theorem: upper bounds and related results*. Charles Univ., 2004.

[42] H. S. Wilf, *Generatingfunctionology*. Elsevier, 2013.

[43] D. Berend and T. Tassa, "Improved bounds on Bell numbers and on moments of sums of random variables," *Probability and Mathematical Statistics*, vol. 30, no. 2, pp. 185–205, 2010.

[44] B. Balasundaram, S. S. Chandramouli, and S. Trukhanov, "Approximation algorithms for finding and partitioning unit-disk graphs into co-k-plexes," *Optimization Letters*, vol. 4, no. 3, pp. 311–320, 2010.

[45] M. Aigner, "Turán's graph theorem," *The American Mathematical Monthly*, vol. 102, pp. 808–816, 1995.

[46] Y. Caro, "New results on the independence number," tech. rep., Technical Report, Tel-Aviv University, 1979.

[47] V. Wei, "A lower bound on the stability number of a simple graph," tech. rep., Bell Laboratories Technical Memorandum 81-11217-9, Murray Hill, NJ, 1981.

[48] S. M. Selkow, "A probabilistic lower bound on the independence number of graphs," *Discrete Mathematics*, vol. 132, no. 1-3, pp. 363–365, 1994.

[49] R. Carraghan and P. M. Pardalos, "An exact algorithm for the maximum clique problem," *Operations Research Letters*, vol. 9, no. 6, pp. 375–382, 1990.

[50] P. R. Östergård, "A fast algorithm for the maximum clique problem," *Discrete Applied Mathematics*, vol. 120, no. 1-3, pp. 197–207, 2002.

[51] S. Trukhanov, C. Balasubramaniam, B. Balasundaram, and S. Butenko, "Algorithms for detecting optimal hereditary structures in graphs, with application to clique relaxations," *Computational Optimization and Applications*, vol. 56, no. 1, pp. 113–130, 2013.

[52] G. Verfaillie, M. Lemaître, and T. Schiex, "Russian doll search for solving constraint optimization problems," in *AAAI/IAAI, Vol. 1*, pp. 181–187, 1996.

[53] P. Meseguer and M. Sánchez, "Tree-based Russian doll search: Preliminary results," in *Proceedings of the CP'00 Workshop on Soft Constraints*, Citeseer, 2000.

[54] P. Meseguer and M. Sánchez, "Specializing russian doll search," in *International Conference on Principles and Practice of Constraint Programming*, pp. 464–478, Springer, 2001.

[55] P. Mesegue, M. Sánchez, and G. Verfaillie, "Opportunistic specialization in russian doll search," in *International Conference on Principles and Practice of Constraint Programming*, pp. 264–279, Springer, 2002.

[56] V. Vaskelainen *et al.*, "Russian doll search algorithms for discrete optimization problems," 2010.

[57] P. R. Östergård and V. P. Vaskelainen, "Russian doll search for the Steiner triple covering problem," *Optimization Letters*, vol. 5, no. 4, pp. 631–638, 2011.

[58] D. A. Bader, W. E. Hart, and C. A. Phillips, "Parallel algorithm design for branch and bound," in *Tutorials on Emerging Methodologies and Applications in Operations Research*, pp. 5–1–5–44, Springer, 2005.

[59] D. E. Culler, J. P. Singh, and A. Gupta, *Parallel computer architecture: a hardware/software approach*. Gulf Professional Publishing, 1999.

[60] J. Dongarra, I. Foster, G. Fox, W. Gropp, K. Kennedy, L. Torczon, and A. White, *Sourcebook of parallel computing*, vol. 3003. Morgan Kaufmann Publishers San Francisco, 2003.

[61] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to parallel computing: design and analysis of algorithms*, vol. 400. Benjamin/Cummings Redwood City, 1994.

[62] L. Jordan and G. Alaghband, *Fundamentals of parallel processing*. Prentice Hall Professional Technical Reference, 2002.

[63] J. JáJá, *An introduction to parallel algorithms*, vol. 17. Addison-Wesley Reading, 1992.

[64] R. Miller and L. Boxer, *Algorithms sequential & parallel: A unified approach*. Cengage Learning, 2012.

[65] J. H. Reif, *Synthesis of parallel algorithms*. Morgan Kaufmann Publishers Inc., 1993.

[66] S. Fortune and J. Wyllie, "Parallelism in random access machines," in *Proceedings of the tenth annual ACM symposium on Theory of computing*, pp. 114–118, ACM, 1978.

[67] L. Dagum and R. Menon, "OpenMP: an industry standard API for shared-memory programming," *IEEE computational science and engineering*, vol. 5, no. 1, pp. 46–55, 1998.

[68] R. Chandra, L. Dagum, D. Kohr, D. Maydan, R. Menon, and J. McDonald, *Parallel programming in OpenMP*. Morgan Kaufmann, 2001.

[69] M. J. Quinn, "Parallel programming," *TMH CSE*, vol. 526, 2003.

[70] University of Tennessee, "PVM: Parallel virtual machine," 2004.

[71] M. Snir, S. Otto, S. Huss-Lederman, J. Dongarra, and D. Walker, *MPI–the Complete Reference: The MPI core*, vol. 1. MIT press, 1998.

[72] W. D. Gropp, W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: portable parallel programming with the message-passing interface*, vol. 1. MIT press, 1999.

[73] R. H. Bisseling, *Parallel Scientific Computation: A structured approach using BSP and MPI*. Oxford University Press on Demand, 2004.

[74] B. Wilkinson, C. Allen, *et al.*, *Parallel programming: techniques and applications using networked workstations and parallel computers*. Upper Saddle River, NJ: Pearson/Prentice Hall, 2005.

[75] L. Barreto and M. Bauer, "Parallel Branch and Bound Algorithm – A comparison between serial, OpenMP and MPI implementations," in *Journal of Physics: Conference Series*, vol. 256, p. 012018, IOP Publishing, 2010.

[76] R. C. Corrêa, P. Michelon, B. L. Cun, T. Mautor, and D. D. Donne, "A bit-parallel russian dolls search for a maximum cardinality clique in a graph," *arXiv preprint arXiv:1407.1209*, 2014.

[77] B. Nogueira and R. G. S. Pinheiro, "A GPU based local search algorithm for the unweighted and weighted maximum s-plex problems," *Annals of Operations Research*, Feb 2019.

[78] O. Yezerska, S. Butenko, and V. L. Boginski, "Detecting robust cliques in graphs subject to uncertain edge failures," *Annals of Operations Research*, vol. 262, no. 1, pp. 109–132, 2018.

[79] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference*, pp. 483–485, ACM, 1967.

[80] J. Pardalos and M. Resende, "On maximum clique problems in very large graphs," *DIMACS series*, vol. 50, pp. 119–130, 1999.

[81] S. Shahinpour, S. Shirvani, Z. Ertem, and S. Butenko, "Scale reduction techniques for computing maximum induced bicliques," *Algorithms*, vol. 10, no. 4, p. 113, 2017.

[82] K. I. Appel and W. Haken, *Every planar map is four colorable*, vol. 98. American Mathematical Soc., 1989.

[83] K. Wagner, "Über eine eigenschaft der ebenen komplexe," *Mathematische Annalen*, vol. 114, no. 1, pp. 570–590, 1937.

[84] N. Robertson, P. Seymour, and R. Thomas, "Hadwiger's conjecture for K_6-free graphs," *Combinatorica*, vol. 13, no. 3, pp. 279–361, 1993.

[85] B. Bollobás, P. A. Catlin, and P. Erdös, "Hadwiger's conjecture is true for almost every graph.," *Eur. J. Comb.*, vol. 1, no. 3, pp. 195–199, 1980.

[86] P. Seymour, "Hadwiger's conjecture," 2016. `https://web.math.princeton.edu/~pds/papers/hadwiger/paper.pdf`.

[87] N. Alon, A. Lingas, and M. Wahlen, "Approximating the maximum clique minor and some subgraph homeomorphism problems," *Theoretical Computer Science*, vol. 374, no. 1-3, pp. 149–158, 2007.

[88] D. Eppstein, "Finding large clique minors is hard.," *J. Graph Algorithms Appl.*, vol. 13, no. 2, pp. 197–204, 2009.

[89] M. Fischetti, M. Leitner, I. Ljubić, M. Luipersbeck, M. Monaci, M. Resch, D. Salvagnin, and M. Sinnl, "Thinning out steiner trees: a node-based model for uniform edge costs," *Mathematical Programming Computation*, vol. 9, no. 2, pp. 203–229, 2017.

[90] Y. Wang, A. Buchanan, and S. Butenko, "On imposing connectivity constraints in integer programs," *Mathematical Programming*, vol. 166, no. 1-2, pp. 241–271, 2017.

[91] A. Buchanan, E. Lykhovyd, and H. Validi, "Imposing contiguity constraints in political districting models," *Working paper*, 2019.

[92] J. Oehrlein and J.-H. Haunert, "A cutting-plane method for contiguity-constrained spatial aggregation," *Journal of Spatial Information Science*, vol. 2017, no. 15, pp. 89–120, 2017.

[93] T. Shirabe, "Districting modeling with exact contiguity constraints," *Environment and Planning B: Planning and Design*, vol. 36, no. 6, pp. 1053–1066, 2009.

[94] V. Kaibel, "Extended formulations in combinatorial optimization," *arXiv preprint arXiv:1104.1023*, 2011.

[95] L. R. Ford and D. R. Fulkerson, *Flows in networks*. Princeton University Press, 1962.

[96] S. Butenko, P. Festa, and P. Pardalos, "On the chromatic number of graphs," *Journal of Optimization Theory and Applications*, vol. 109, no. 1, pp. 69–83, 2001.

[97] J. Hopcroft and R. Tarjan, "Algorithm 447: efficient algorithms for graph manipulation," *Communications of the ACM*, vol. 16, no. 6, pp. 372–378, 1973.

[98] A. M. Geoffrion, "Lagrangean relaxation for integer programming," in *Approaches to integer programming*, pp. 82–114, Springer, 1974.

[99] D. S. Johnson, "The NP-completeness column: an ongoing guide," *Journal of Algorithms*, vol. 6, no. 3, pp. 434–451, 1985.

[100] G. R. Grimmett and C. J. McDiarmid, "On colouring random graphs," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 77, pp. 313–324, Cambridge University Press, 1975.

[101] A. Buchanan, J. L. Walteros, S. Butenko, and P. M. Pardalos, "Solving maximum clique in sparse graphs: an $O(nm + n2^{d/4})$ algorithm for d-degenerate graphs," *Optimization Letters*, vol. 8, no. 5, pp. 1611–1617, 2014.

APPENDIX A

RUSSIAN DOLL SEARCH COMPUTATIONAL RESULTS

A.1 IUC and MPC performance

The following two tables demonstrate improvements for IUC and MPC (complement of IUC). The columns are:

- *Graph* : graph file name;
- RDS_e *obj.* : size of the best solution in [1];
- RDS_e *time* : time in seconds in [1];
- RDS_{ver} *obj.* : size of the best solution of sequential RDS, Algorithm 2, with verifier described in Algorithm 7;
- RDS_{ver} *time* : time in seconds of sequential RDS, Algorithm 2, with verifier described in Algorithm 7;
- RDS_{par} *obj.* : size of the best solution of parallel RDS, Algorithm 9, and verifier described in Algorithm 7;
- RDS_{par} *time* : time in seconds of parallel RDS, Algorithm 9, with verifier described in Algorithm 7.

All computations were performed on 15 threads. The PC features are Intel(R) Xeon(R) CPU E5620 at 2.40GHz, 8 GB RAM, compiled with GNU C++ compiler version 4.9.2 with -O2 optimization option. The time limit was 10 hours and denoted as †.

Table A.1: Comparison of RDS performance on maximum IUC problem against [1].

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
brock200_1	21	201.54	21	204.113	21	31.39
brock200_2	15	29.74	15	34.73	15	4.56
brock200_3	15	4.24	15	5.01	15	0.6
brock200_4	17	5.46	17	3.66	17	1.01
brock400_1	23	†	23 (86)	†	27	33165
brock400_2	23	†	24 (74)	†	29	9544.67
brock400_3	23	†	29 (26)	†	31	4451.27
brock400_4	33	31,345.19	33	12,534.50	33	2344.3
brock800_1	20	†	20 (231)	†	21(101)	†
brock800_2	20	†	20 (244)	†	22 (106)	†
brock800_3	20	†	20 (219)	†	22 (66)	†
c-fat200-1	130	0.87	130	0.09	130	0.13
c-fat200-2	134	1.07	134	0.09	134	0.14
c-fat200-5	115	0.51	115	0.05	115	0.08
c-fat500-10	313	67.40	313	2.59	313	2.85
c-fat500-1	332	90.31	332	3.23	332	3.57
c-fat500-2	326	83.71	326	2.99	326	3.36
c-fat500-5	313	67.58	313	2.53	313	2.82
DSJC500_5	17	†	17 (8)	†	17	6304.72
gen200_p0.9_44	34	†	36 (58)	†	37 (48)	†
gen200_p0.9_55	42	†	45 (29)	†	55	10254.20
gen400_p0.9_55	36	†	36 (267)	†	38	†
gen400_p0.9_65	33	†	36 (254)	†	38 (246)	†
gen400_p0.9_75	36	†	36 (258)	†	37 (249)	†
hamming6-2	32	0.01	32	0.00	32	0.00
hamming6-4	16	0.11	16	0.11	16	0.04
hamming8-2	128	0.99	128	0.11	128	0.17
hamming8-4	16	60.16	16	40.15	16	7.95
hamming10-2	512	1,009.13	512	32.45	512	26.98

Continued on next page

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
hamming10-4	18	†	18 (672)	†	18 (639)	†
johnson8-2-4	7	0.01	7	0.00	7	0.00
johnson8-4-4	14	0.01	14	0.00	14	0.00
johnson16-2-4	15	1.69	15	1.09	15	0.24
johnson32-2-4	31	†	31 (251)	†	31 (230)	†
MANN_a9	16	0.01	16	0.00	16	0.00
MANN_a27	119	†	119 (24)	†	119 (24)	†
keller4	15	3.02	15	2.00	15	0.45
p_hat300-1	33	†	36 (127)	†	36 (115)	†
p_hat300-2	32	†	33 (1)	†	33	12688.70
p_hat300-3	31	†	33 (37)	†	36	33545.00
p_hat500-1	34	†	37 (313)	†	38 (298)	†
p_hat500-2	35	†	37 (154)	†	38 (130)	†
p_hat500-3	36	†	38 (230)	†	39 (218)	†
p_hat700-1	33	†	35 (543)	†	36 (533)	†
p_hat700-2	37	†	37 (404)	†	42 (377)	†
p_hat700-3	35	†	38 (406)	†	38 (394)	†
san200_0.7_1	30	5,026.53	30	1,878.41	30	584.52
san200_0.7_2	17	†	17 (29)	†	18	35417.50
san200_0.9_1	37	†	38 (88)	†	39 (84)	†
san200_0.9_2	36	†	51 (25)	†	60	10386.20
san200_0.9_3	33	†	33 (70)	†	34 (58)	†
san400_0.5_1	29	†	30 (130)	†	33 (89)	†
san400_0.7_1	22	†	22 (255)	†	22 (253)	†
san400_0.7_2	18	†	19 (223)	†	20 (203)	†
san400_0.7_3	16	†	16 (198)	†	17 (164)	†
san400_0.9_1	42	†	44 (262)	†	44 (259)	†
sanr200_0.7	18	21.55	18	16.34	18	3.36
sanr200_0.9	35	†	35 (62)	†	36 (51)	†
sanr400_0.5	17	12,622.94	17	9,443.44	17	1261.48

Continued on next page

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
sanr400_0.7	21	†	21	33,371.8	21	4188.17

Table A.2: Comparison of RDS performance on maximum MPC problem against [1].

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
brock200_1	24	†	24 (31)	†	28	23251
brock200_2	14	37.31	14	26.84	14	5.42
brock200_3	18	775.64	18	486.36	18	102.51
brock200_4	20	5,652.76	20	3050.92	20	587.91
brock400_1	25	†	25 (223)	†	26 (213)	†
brock400_2	27	†	27 (209)	†	27 (195)	†
brock400_3	26	†	26 (218)	†	26 (203)	†
brock400_4	24	†	26 (219)	†	26 (205)	†
brock800_1	20	†	21 (542)	†	22 (483)	†
brock800_2	21	†	21 (550)	†	22 (500)	†
brock800_3	20	†	21 (583)	†	21 (501)	†
c-fat200-1	18	†	18 (103)	†	18 (95)	†
c-fat200-2	24	11.08	24	6.42	24	2.29
c-fat200-5	58	0.02	58	0.01	58	0.01
c-fat500-10	126	0.86	126	0.12	126	0.18
c-fat500-1	40	†	40 (407)	†	40 (406)	†
c-fat500-2	20	†	20 (407)	†	20 (399)	†
c-fat500-5	64	4.83	64	2.84	64	1.42
DSJC500_5	17	†	17 (13)	†	17	7405.23
gen200_p0.9_44	37	†	38 (105)	†	38 (101)	†
gen200_p0.9_55	37	†	37 (111)	†	38 (108)	†
gen400_p0.9_55	37	†	39 (309)	†	41 (303)	†
gen400_p0.9_65	40	†	44 (288)	†	44 (285)	†
gen400_p0.9_75	37	†	38 (302)	†	42 (293)	†

Continued on next page

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
hamming6-2	32	0.01	32	0.00	32	0.00
hamming6-4	14	0.01	14	0.00	14	0.00
hamming8-2	128	2.29	128	0.28	128	0.31
hamming8-4	32	†	32 (12)	†	32	17234.50
hamming10-2	512	2,424.65	512	84.62	512	64.58
hamming10-4	32	†	32 (780)	†	36 (760)	†
johnson8-2-4	7	0.01	7	0.00	7	0.00
johnson8-4-4	16	0.73	16	0.29	16	0.09
johnson16-2-4	15	†	15 (5)	†	15	10146.10
johnson32-2-4	31	†	31 (426)	†	31 (423)	†
MANN_a9	36	0.03	36	0.01	36	0.01
MANN_a27	351	†	351 (21)	†	351 (21)	†
keller4	26	65.85	26	29.12	26	10.17
p_hat300-1	37	†	37 (33)	†	37 (22)	†
p_hat300-2	31	23,452.41	31	7,880	31	2534.82
p_hat300-3	36	†	37 (132)	†	38 (110)	†
p_hat500-1	36	†	38 (191)	†	40 (178)	†
p_hat500-2	36	†	37 (192)	†	38 (161)	†
p_hat500-3	36	†	37 (341)	†	39 (329)	†
p_hat700-1	39	†	42 (402)	†	42 (395)	†
p_hat700-2	35	†	35 (360)	†	36 (305)	†
p_hat700-3	34	†	40 (495)	†	41 (491)	†
san200_0.7_1	105	0.33	105	0.07	105	0.06
san200_0.7_2	134	1.05	134	0.11	134	0.15
san200_0.9_1	125	7.31	125	3.25	125	1.48
san200_0.9_2	105	8,155.61	105	2,774.00	105	979.82
san200_0.9_3	100	848.55	100	283.4	100	123.92
san400_0.5_1	214	10.05	214	0.62	214	0.83
san400_0.7_1	200	7.42	200	0.56	200	0.73
san400_0.7_2	205	8.21	205	0.86	205	0.76

Continued on next page

Graph	RDS_e obj.	RDS_e time	RDS_{ver} obj.	RDS_{ver} time	RDS_{par} obj.	RDS_{par} time
san400_0.7_3	216	10.35	216	0.90	216	0.90
san400_0.9_1	200	1,036.64	200	366.11	200	136.85
sanr200_0.7	22	†	22	25519.9	22	4917.71
sanr200_0.9	36	†	37 (106)	†	37 (103)	†
sanr400_0.5	16	17,000.38	16	11341.10	16	1565.13
sanr400_0.7	22	†	23 (186)	†	24 (155)	†

A.2 Biclique performance

The next table presents results of parallel RDS performance with biclique verification procedure. The columns are:

- $|V|$ - number of vertices;
- $|E|$ - number of edges;
- $\rho(G)$ - graph density in percents;
- $\omega(G)$ - clique number;
- $bc(G)$ - the best biclique found by Algorithm 9;
- i^* - the last $\mu[i^*]$ computed by Algorithm 9;
- *Time* - time taken in seconds; † means time limit of 600 seconds.

Table A.3: RDS performance for the maximum biclique problem.

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
brock200_1.clq	200	14,834	74.54	21	11	0	0.38
brock200_2.clq	200	9,876	49.62	12	13	0	0.51
brock200_3.clq	200	12,048	60.54	15	12	0	0.45

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
brock200_4.clq	200	13,089	65.77	17	12	0	0.41
brock400_1.clq	400	59,723	74.84	27	13	0	13.41
brock400_2.clq	400	59,786	74.92	29	13	0	9.88
brock400_3.clq	400	59,681	74.78	31	13	0	12.29
brock400_4.clq	400	59,765	74.89	33	13	0	12.59
brock800_1.clq	800	207,505	64.92	23	15	180	†
brock800_2.clq	800	208,166	65.13	24	15	186	†
brock800_3.clq	800	207,333	64.87	25	15	227	†
brock800_4.clq	800	207,643	64.97	26	15	178	†
c-fat200-1.clq	200	1,534	7.70	12	18	117	†
c-fat200-2.clq	200	3,235	16.25	24	9	11	†
c-fat200-5.clq	200	8,473	42.57	58	3	0	0.20
c-fat500-1.clq	500	4,459	3.57	14	40	409	†
c-fat500-10.clq	500	46,627	37.37	126	4	0	2.96
c-fat500-2.clq	500	9,139	7.32	26	20	419	†
c-fat500-5.clq	500	23,191	18.59	64	8	251	†
hamming10-2.clq	1,024	518,656	99.02	512	4	0	3.80
hamming10-4.clq	1,024	434,176	82.89	40	34	708	†
hamming6-2.clq	64	1,824	90.47	32	4	0	0.00
hamming6-4.clq	64	704	34.92	4	14	0	0.01
hamming8-2.clq	256	31,616	96.86	128	4	0	0.06
hamming8-4.clq	256	20,864	63.92	16	32	0	111.82
johnson16-2-4.clq	120	5,460	76.47	8	15	0	1.41
johnson32-2-4.clq	496	107,880	87.87	16	31	407	†
johnson8-2-4.clq	28	210	55.55	4	7	0	0.02
johnson8-4-4.clq	70	1,855	76.81	14	10	0	0.03
keller4.clq	171	9,435	64.91	11	19	0	1.43
keller5.clq	776	225,990	75.15	27	32	342	†
keller6.clq	3,361	4,619,898	81.81	59	32	2,944	†
MANN_a27.clq	378	70,551	99.01	126	6	0	0.13

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
MANN_a45.clq	1,035	533,115	99.63	345	6	0	2.39
MANN_a81.clq	3,321	5,506,380	99.88	1,100	6	0	94.17
MANN_a9.clq	45	918	92.72	16	6	0	0.19
p_hat1000-1.clq	1,000	122,253	24.47	10	34	734	†
p_hat1000-2.clq	1,000	244,799	49.00	46	34	483	†
p_hat1000-3.clq	1,000	371,746	74.42	68	15	240	†
p_hat1500-1.clq	1,500	284,923	25.34	12	39	1,214	†
p_hat1500-2.clq	1,500	568,960	50.60	65	36	1,018	†
p_hat1500-3.clq	1,500	847,244	75.36	94	14	773	†
p_hat300-1.clq	300	10,933	24.37	8	36	63	†
p_hat300-2.clq	300	21,928	48.89	25	27	0	4.53
p_hat300-3.clq	300	33,390	74.44	36	13	0	2.47
p_hat500-1.clq	500	31,569	25.30	9	35	241	†
p_hat500-2.clq	500	62,946	50.45	36	36	4	†
p_hat500-3.clq	500	93,800	75.19	50	14	0	31.30
p_hat700-1.clq	700	60,999	24.93	11	38	443	†
p_hat700-2.clq	700	121,728	49.75	44	38	250	†
p_hat700-3.clq	700	183,010	74.80	62	14	3	†
san1000.clq	1,000	250,500	50.15	15	134	0	0.42
san200_0.7_1.clq	200	13,930	70	30	15	0	0.03
san200_0.7_2.clq	200	13,930	70	18	24	0	0.02
san200_0.9_1.clq	200	17,910	90	70	8	0	0.03
san200_0.9_2.clq	200	17,910	90	60	8	0	0.05
san200_0.9_3.clq	200	17,910	90	44	10	0	0.03
san400_0.5_1.clq	400	39,900	50	13	62	0	0.07
san400_0.7_1.clq	400	55,860	70	40	20	0	1.15
san400_0.7_2.clq	400	55,860	70	30	28	0	0.40
san400_0.7_3.clq	400	55,860	70	22	38	0	0.17
san400_0.9_1.clq	400	71,820	90	100	10	0	0.67
sanr200_0.7.clq	200	13,868	69.68	18	12	0	0.35

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
sanr200_0.9.clq	200	17,863	89.76	42	8	0	0.06
sanr400_0.5.clq	400	39,984	50.10	13	14	0	53.77
sanr400_0.7.clq	400	55,869	70.01	21	14	0	22.24
adjnoun.graph	112	425	6.83	5	53	0	93.15
as-22july06.graph	22,963	48,436	0.01	17	917	22,031	†
astro-ph.graph	16,706	121,251	0.08	57	98	15,866	†
celegans_metabolic.graph	453	2,025	1.97	9	62	358	†
celegansneural.graph	297	2,148	4.88	8	71	166	†
chesapeake.graph	39	170	22.94	5	17	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	99	16,083	†
cond-mat-2003.graph	31,163	120,029	0.02	25	255	30,074	†
cond-mat-2005.graph	40,421	175,691	0.02	30	223	39,246	†
dolphins.graph	62	159	8.40	5	28	0	0.18
email.graph	1,133	5,451	0.85	12	114	996	†
football.graph	115	613	9.35	9	21	11	†
hep-th.graph	8,361	15,751	0.04	24	149	7,415	†
jazz.graph	198	2,742	14.05	30	30	60	†
karate.graph	34	78	13.90	5	20	0	0.01
lesmis.graph	77	254	8.68	10	34	0	0.02
memplus.graph	17,758	54,196	0.03	97	4,225	9,307	†
netscience.graph	1,589	2,742	0.21	20	86	1,250	†
PGPgiantcompo.graph	10,680	24,316	0.04	25	3,361	7,293	†
polblogs.graph	1,490	16,715	1.50	20	98	1,090	†
polbooks.graph	105	441	8.07	6	43	0	3.13
power.graph	4,941	6,594	0.05	6	62	4,831	†
Cit-HepTh.txt	27,769	352,285	0.09	23	193	27,562	†
Wiki-Vote.txt	7,115	100,762	0.39	17	249	6,817	†
1-FullIns_3.col	30	100	22.98	3	14	0	0.00
1-FullIns_4.col	93	593	13.86	3	45	0	0.05
1-FullIns_5.col	282	3,247	8.19	3	95	111	†

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
1-Insertions_4.col	67	232	10.49	2	32	0	0.01
1-Insertions_5.col	202	1,227	6.04	2	68	67	†
1-Insertions_6.col	607	6,337	3.44	2	203	338	†
2-FullIns_3.col	52	201	15.15	4	25	0	0.02
2-FullIns_4.col	212	1,621	7.24	4	105	37	†
2-FullIns_5.col	852	12,201	3.36	4	214	582	†
2-Insertions_3.col	37	72	10.81	2	18	0	0.161
2-Insertions_4.col	149	541	4.90	2	74	0	210.03
2-Insertions_5.col	597	3,936	2.21	2	150	377	†
3-FullIns_3.col	80	346	10.94	5	37	0	0.07
3-FullIns_4.col	405	3,524	4.30	5	82	253	†
3-FullIns_5.col	2,030	33,751	1.63	5	407	1,558	†
3-Insertions_3.col	56	110	7.14	2	27	0	0.00
3-Insertions_4.col	281	1,046	2.65	2	57	171	†
3-Insertions_5.col	1,406	9,695	0.98	2	282	1074	†
4-FullIns_3.col	114	541	8.39	6	55	0	0.23
4-FullIns_4.col	690	6,650	2.79	6	116	500	†
4-FullIns_5.col	4,146	77,305	0.9	6	692	3391	†
4-Insertions_3.col	79	156	5.06	2	39	0	0.06
4-Insertions_4.col	475	1,795	1.59	2	80	339	†
5-FullIns_3.col	154	792	6.72	7	72	0	5.72
5-FullIns_4.col	1,085	11,395	1.93	7	156	857	†
abb313GPIA.col	1,557	53,356	4.40	8	313	0	0.72
anna.col	138	493	5.21	11	56	39	†
ash331GPIA.col	662	4,181	1.91	3	194	105	†
ash608GPIA.col	1,216	7,844	1.06	3	101	929	†
ash958GPIA.col	1,916	12,506	0.68	3	192	1365	†
C2000.5.col	2,000	999,836	50.01	16	15	1433	†
C4000.5.col	4,000	4,000,268	50.01	18	15	3441	†
david.col	87	406	10.85	11	36	0	107.05

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
DSJC1000.1.col	1,000	49,629	9.936	6	33	886	†
DSJC1000.5.col	1,000	249,826	50.02	15	16	391	†
DSJC1000.9.col	1,000	449,449	89.98	68	12	0	122.52
DSJC125.1.col	125	736	9.49	4	34	1	†
DSJC125.5.col	125	3,891	50.20	10	11	0	0.07
DSJC125.9.col	125	6,961	89.81	34	8	0	0.00
DSJC250.1.col	250	3,218	10.33	4	35	112	†
DSJC250.5.col	250	15,668	50.33	12	13	0	2.06
DSJC250.9.col	250	27,897	89.62	43	9	0	0.12
DSJC500.1.col	500	12,458	9.98	5	34	368	†
DSJC500.5.col	500	62,624	50.2	13	15	0	203.02
DSJC500.9.col	500	112,437	90.13	56	10	0	2.60
DSJR500.1.col	500	3,555	2.85	11	38	421	†
DSJR500.1c.col	500	121,275	97.21	83	25	0	0.08
DSJR500.5.col	500	58,862	47.18	122	7	0	4.17
flat1000_50_0.col	1,000	245,000	49.04	15	16	397	†
flat1000_60_0.col	1,000	245,830	49.21	15	15	426	†
flat1000_76_0.col	1,000	246,708	49.39	15	15	436	†
flat300_20_0.col	300	21,375	47.65	11	15	0	1.63
flat300_26_0.col	300	21,633	48.23	11	13	0	7.02
flat300_28_0.col	300	21,695	48.37	12	14	0	6.24
fpsol2.i.1.col	496	11,654	9.49	65	280	88	†
fpsol2.i.2.col	451	8,691	8.56	30	245	68	†
fpsol2.i.3.col	425	8,688	9.64	30	222	68	†
games120.col	120	638	8.93	9	22	16	†
homer.col	561	1,628	1.03	13	101	433	†
huck.col	74	301	11.14	11	27	0	6.16
inithx.i.1.col	864	18,707	5.01	54	565	71	†
inithx.i.2.col	645	13,979	6.73	31	365	61	†
inithx.i.3.col	621	13,969	7.25	31	360	61	†

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
jean.col	80	254	8.03	10	38	0	0.86
latin_square_10.col	900	307,350	75.97	90	18	0	63.82
le450_15a.col	450	8,168	8.08	15	37	329	†
le450_15b.col	450	8,169	8.08	15	41	332	†
le450_15c.col	450	16,680	16.51	15	27	290	†
le450_15d.col	450	16,750	16.58	15	27	285	†
le450_25a.col	450	8,260	8.17	25	41	343	†
le450_25b.col	450	8,263	8.17	25	33	352	†
le450_25c.col	450	17,343	17.16	25	31	281	†
le450_25d.col	450	17,425	17.24	25	28	294	†
le450_5a.col	450	5,714	5.65	5	34	360	†
le450_5b.col	450	5,734	5.67	5	36	354	†
le450_5c.col	450	9,803	9.70	5	34	324	†
le450_5d.col	450	9,757	9.65	5	33	329	†
miles1000.col	128	3,216	39.56	42	8	0	0.05
miles1500.col	128	5,198	63.95	73	5	0	0.01
miles250.col	128	387	4.76	8	35	39	†
miles500.col	128	1,170	14.39	20	18	3	†
miles750.col	128	2,113	25.99	31	12	0	1.44
mug100_1.col	100	166	3.35	3	33	0	244.96
mug100_25.col	100	166	3.35	3	31	7	†
mug88_1.col	88	146	3.81	3	29	0	185.40
mug88_25.col	88	146	3.814	3	29	0	112.48
mulsol.i.1.col	197	3,925	20.33	49	100	0	3.11
mulsol.i.2.col	188	3,885	22.10	31	90	34	†
mulsol.i.3.col	184	3,916	23.26	31	86	16	†
mulsol.i.4.col	185	3,946	23.18	31	86	34	†
mulsol.i.5.col	186	3,973	23.09	31	88	34	†
myciel3.col	11	20	36.36	2	6	0	0.00
myciel4.col	23	71	28.06	2	12	0	0.00

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
myciel5.col	47	236	21.83	2	24	0	0.02
myciel6.col	95	755	16.90	2	48	0	0.33
myciel7.col	191	2,360	13.00	2	96	0	3.34
qg.order30.col	900	26,100	6.45	30	30	0	0.13
qg.order40.col	1,600	62,400	4.87	40	40	0	0.37
qg.order60.col	3,600	212,400	3.27	60	60	0	2.07
queen10_10.col	100	1,470	29.69	10	10	0	0.06
queen11_11.col	121	1,980	27.27	11	11	0	0.04
queen12_12.col	144	2,596	25.21	12	12	0	0.05
queen13_13.col	169	3,328	23.44	13	13	0	0.10
queen14_14.col	196	4,186	21.90	14	14	0	0.23
queen15_15.col	225	5,180	20.55	15	15	0	0.78
queen16_16.col	256	6,320	19.36	16	16	0	1.66
queen5_5.col	25	160	53.33	5	8	0	0.00
queen6_6.col	36	290	46.03	6	8	0	0.00
queen7_7.col	49	476	40.47	7	8	0	0.01
queen8_12.col	96	1,368	30	12	8	0	0.05
queen8_8.col	64	728	36.11	8	8	0	0.00
queen9_9.col	81	1,056	32.59	9	9	0	0.01
r1000.1.col	1,000	14,378	2.87	20	38	920	†
r1000.1c.col	1,000	485,090	97.11	91	43	0	0.67
r1000.5.col	1,000	238,267	47.70	234	7	0	194.07
r125.1.col	125	209	2.69	5	39	48	†
r125.1c.col	125	7,501	96.78	46	13	0	0.00
r125.5.col	125	3,838	49.52	36	6	0	0.02
r250.1.col	250	867	2.78	8	36	179	†
r250.1c.col	250	30,227	97.11	63	16	0	0.09
r250.5.col	250	14,849	47.70	65	6	0	0.30
school1.col	385	19,095	25.83	14	32	137	†
school1_nsh.col	352	14,612	23.65	14	31	114	†

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	$bc(G)$	i^*	Time
wap01a.col	2,368	110,871	3.95	41	17	2167	†
wap02a.col	2,464	111,742	3.68	40	29	2379	†
wap03a.col	4,730	286,722	2.56	40	20	4552	†
wap04a.col	5,231	294,902	2.15	40	56	5093	†
wap05a.col	905	43,081	10.53	50	15	626	†
wap06a.col	947	43,571	9.72	40	24	743	†
wap07a.col	1,809	103,368	6.32	40	19	1558	†
wap08a.col	1,870	104,176	5.96	40	29	1763	†
will199GPIA.col	701	6,772	2.76	6	110	321	†
zeroin.i.1.col	211	4,100	18.50	49	120	0	6.22
zeroin.i.2.col	211	3,541	15.98	30	127	0	103.99
zeroin.i.3.col	206	3,540	16.76	30	123	0	72.30

A.3 Plex and defective performance

The next tables provide computational experiments on standard benchmark instances considered in [2]. We could not entirely replicate the performance of previous algorithm, which is not in public access and assume there were undisclosed algorithmic tricks, making their algorithm efficient on huge sparse instances. Thus, we compare our parallel shared-memory version to our sequential version, in addition to results in [2]. The columns are:

- $|V|$ - number of vertices;
- $|E|$ - number of edges;
- $\rho(G)$ - graph density in percents;
- $\omega(G)$ - clique number;
- V_{red} - size of the reduced (peeled) instance;
- opt_G - the best solution found in [2];

- opt_{seq} - the best solution found by our sequential algorithm;
- opt_{par} - the best solution found by our shared-memory parallel algorithm;
- i^* - the last $\mu[i^*]$ computed by RDS;
- *Time* - time taken in seconds; † means time limit of 600 seconds.

Table A.4: RDS performance for 2-plex.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	23	†	23	64	†	23	51	†
brock200_2.clq	200	9,876	49.63	12	200	13	8.63	13	0	9.40	13	0	2.37
brock200_3.clq	200	12,048	60.54	15	200	17	166.44	17	0	169.00	17	0	40.69
brock200_4.clq	200	13,089	65.77	17	200	19	†	19	12	†	20	0	289.34
brock400_1.clq	400	59,723	74.84	27	400	23	†	24	260	†	25	245	†
brock400_2.clq	400	59,786	74.92	29	400	22	†	23	264	†	23	255	†
brock400_3.clq	400	59,681	74.79	31	400	23	†	23	263	†	23	251	†
brock400_4.clq	400	59,765	74.89	33	400	23	†	23	261	†	23	251	†
brock800_1.clq	800	207,505	64.93	23	800	19	†	19	600	†	19	576	†
brock800_2.clq	800	208,166	65.13	24	800	19	†	19	613	†	19	591	†
brock800_3.clq	800	207,333	64.87	25	800	19	†	19	605	†	20	572	†
brock800_4.clq	800	207,643	64.97	26	800	19	†	19	603	†	20	577	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.01	12	0	0.01	12	0	0.01
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.02	24	0	0.00	24	0	0.01
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.01	58	0	0.00	58	0	0.01
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.01	14	0	0.04	14	0	0.03
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.03	126	0	0.02	126	0	0.04
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.01	26	0	0.03	26	0	0.03
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.01	64	0	0.02	64	0	0.03
hamming10-2.clq	1,024	518,656	99.02	512	1,024	512	18.67	512	0	3.59	512	0	3.35

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
hamming10-4.clq	1,024	434,176	82.89	40	1,024	22	†	22	699	†	22	698	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.01	32	0	0.00	32	0	0.00
hamming6-4.clq	64	704	34.92	4	64	6	0.01	6	0	0.00	6	0	0.00
hamming8-2.clq	256	31,616	96.86	128	256	128	0.08	128	0	0.03	128	0	0.04
hamming8-4.clq	256	20,864	63.92	16	256	16	7.88	16	0	11.02	16	0	1.95
johnson16-2-4.clq	120	5,460	76.47	8	120	10	†	10	12	†	10	0	†
johnson32-2-4.clq	496	107,880	87.88	16	496	21	†	21	229	†	21	229	†
johnson8-2-4.clq	28	210	55.56	4	28	5	0.01	5	0	0.00	5	0	0.00
johnson8-4-4.clq	70	1,855	76.81	14	70	14	0.02	14	0	0.02	14	0	0.01
keller4.clq	171	9,435	64.91	11	171	15	101.46	15	0	97.81	15	0	35.31
keller5.clq	776	225,990	75.16	27	776	17	†	17	579	†	19	541	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	17	†	17	3,164	†	19	3,127	†
MANN_a27.clq	378	70,551	99.02	126	378	235	†	235	25	†	235	25	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	661	†	661	43	600.01	661	43	600.01
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	1,660	†	2,161	79	600.07	2,161	79	600.07
MANN_a9.clq	45	918	92.73	16	45	26	0.05	26	0	0.02	26	0	0.02
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	13	566.33	13	3	†	13	0	118.68
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	31	†	31	754	†	35	688	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	33	†	33	858	†	34	850	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	13	†	13	562	†	14	231	600.01
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	27	†	27	1,288	†	30	1,246	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	34	†	34	1,375	†	35	1,364	†
p_hat300-1.clq	300	10,933	24.38	8	300	10	0.76	10	0	0.82	10	0	0.22
p_hat300-2.clq	300	21,928	48.89	25	300	29	†	29	35	†	29	19	†
p_hat300-3.clq	300	33,390	74.45	36	300	31	†	31	165	†	32	159	†
p_hat500-1.clq	500	31,569	25.31	9	500	12	13.04	12	0	13.72	12	0	3.27
p_hat500-2.clq	500	62,946	50.46	36	500	34	†	35	222	†	36	214	†
p_hat500-3.clq	500	93,800	75.19	50	500	35	†	35	367	†	35	359	†
p_hat700-1.clq	700	60,999	24.93	11	700	13	47.38	13	0	67.85	13	0	12.52
p_hat700-2.clq	700	121,728	49.76	44	700	31	†	31	426	†	33	392	†
p_hat700-3.clq	700	183,010	74.81	62	700	33	†	33	552	†	35	534	†
san1000.clq	1,000	250,500	50.15	15	1,000	16	†	16	909	†	16	908	†
san200_0.7_1.clq	200	13,930	70.00	30	200	29	†	29	110	†	29	110	†
san200_0.7_2.clq	200	13,930	70.00	18	200	24	†	24	122	†	24	121	†
san200_0.9_1.clq	200	17,910	90.00	70	200	67	†	67	71	†	67	71	†
san200_0.9_2.clq	200	17,910	90.00	60	200	42	†	42	107	†	42	106	†
san200_0.9_3.clq	200	17,910	90.00	44	200	42	†	42	102	†	42	102	†
san400_0.5_1.clq	400	39,900	50.00	13	400	14	†	14	306	†	14	300	†
san400_0.7_1.clq	400	55,860	70.00	40	400	34	†	34	310	†	34	310	†
san400_0.7_2.clq	400	55,860	70.00	30	400	28	†	28	302	†	28	302	†
san400_0.7_3.clq	400	55,860	70.00	22	400	23	†	23	309	†	23	309	†
san400_0.9_1.clq	400	71,820	90.00	100	400	66	†	75	232	†	75	231	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
sanr200_0.7.clq	200	13,868	69.69	18	200	20	†	20	41	†	21	16	†
sanr200_0.9.clq	200	17,863	89.76	42	200	33	†	34	118	†	34	118	†
sanr400_0.5.clq	400	39,984	50.11	13	400	15	†	15	55	†	15	0	472.10
sanr400_0.7.clq	400	55,869	70.01	21	400	20	†	20	243	†	22	214	†
adjnoun.graph	112	425	6.84	5	89	6	0.02	6	0	0.00	6	0	0.01
as-22july06.graph	22,963	48,436	0.02	17	168	19	0.02	19	0	0.01	19	0	0.01
astro-ph.graph	16,706	121,251	0.09	57	113	57	0.01	57	0	0.00	57	0	0.01
caidaRouterLevel.graph	192,244	609,066	0.00	17	4,021	20	1.97	20	0	36.02	20	0	6.04
celegans_metabolic.graph	453	2,025	1.98	9	92	10	0.02	10	0	0.00	10	0	0.00
celegansneural.graph	297	2,148	4.89	8	251	10	0.02	10	0	0.02	10	0	0.01
chesapeake.graph	39	170	22.94	5	39	7	0.01	7	0	0.00	7	0	0.00
cnr-2000.graph	325,557	2,738,969	0.01	84	86	85	0.01	85	0	0.00	85	0	0.01
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.02	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.02	115	0	0.01	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	18	18	0.01	18	0	0.00	18	0	0.00
cond-mat-2003.graph	31,163	120,029	0.03	25	27	25	0.02	25	0	0.00	25	0	0.00
cond-mat-2005.graph	40,421	175,691	0.02	30	30	30	0.02	30	0	0.00	30	0	0.00
dolphins.graph	62	159	8.41	5	45	6	0.01	6	0	0.00	6	0	0.00
email.graph	1,133	5,451	0.85	12	121	12	0.02	12	0	0.01	12	0	0.00
football.graph	115	613	9.35	9	115	10	0.01	10	0	0.00	10	0	0.00
hep-th.graph	8,361	15,751	0.05	24	24	24	0.01	24	0	0.00	24	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
jazz.graph	198	2,742	14.06	30	30	30	0.02	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	22	6	0.01	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	20	10	0.01	10	0	0.00	10	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.02	97	0	0.01	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	126	29	0.01	29	0	0.01	29	0	0.01
polblogs.graph	1,490	16,715	1.51	20	459	23	0.91	23	0	0.75	23	0	0.38
polbooks.graph	105	441	8.08	6	98	7	0.02	7	0	0.00	7	0	0.00
power.graph	4,941	6,594	0.05	6	36	6	0.01	6	0	0.00	6	0	0.00
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	125	16	0.02	16	0	0.00	16	0	0.00
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	55	19	0.01	19	0	0.00	19	0	0.00
Cit-HepTh.txt	27,769	352,285	0.09	23	7,280	28	206.09	28	639	†	28	0	121.37
Email-EuAll.txt	265,009	364,481	0.00	16	1,883	19	1.70	19	0	3.83	19	0	0.97
Slashdot0811.txt	77,360	469,180	0.02	26	6,139	31	41.37	31	0	201.35	31	0	24.18
Slashdot0902.txt	82,168	504,230	0.02	27	6,086	32	24.40	32	0	149.30	32	0	22.21
soc-Epinions1.txt	75,879	405,740	0.01	23	5,243	28	254.52	28	0	382.19	28	0	74.14
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	0.14	202	0	0.06	202	0	0.06
web-Google.txt	875,713	4,322,051	0.00	44	218	44	†	46	0	0.07	46	0	0.07
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	155	4.67	155	0	6.25	155	0	1.21
web-Stanford.txt	281,903	1,992,636	0.01	61	1,389	59	†	53	310	†	53	310	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,382	21	11.08	21	0	26.03	21	0	7.58

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
1-FullIns_3.col	30	100	22.99	3	30	5	0.01	5	0	0.00	5	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	6	0.02	6	0	0.00	6	0	0.00
1-FullIns_5.col	282	3,247	8.20	3	282	6	0.02	6	0	0.05	6	0	0.02
1-Insertions_4.col	67	232	10.49	2	67	4	0.02	4	0	0.00	4	0	0.00
1-Insertions_5.col	202	1,227	6.04	2	202	4	0.01	4	0	0.01	4	0	0.01
1-Insertions_6.col	607	6,337	3.45	2	607	4	0.16	4	0	0.30	4	0	0.09
2-FullIns_3.col	52	201	15.16	4	52	5	0.02	5	0	0.00	5	0	0.00
2-FullIns_4.col	212	1,621	7.25	4	212	6	0.01	6	0	0.01	6	0	0.01
2-FullIns_5.col	852	12,201	3.37	4	852	7	0.28	7	0	0.68	7	0	0.20
2-Insertions_3.col	37	72	10.81	2	37	4	0.01	4	0	0.00	4	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	4	0.01	4	0	0.01	4	0	0.01
2-Insertions_5.col	597	3,936	2.21	2	597	4	0.03	4	0	0.27	4	0	0.08
3-FullIns_3.col	80	346	10.95	5	80	6	0.02	6	0	0.00	6	0	0.00
3-FullIns_4.col	405	3,524	4.31	5	405	7	0.03	7	0	0.07	7	0	0.03
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	8	1.51	8	0	16.18	8	0	2.63
3-Insertions_3.col	56	110	7.14	2	56	4	0.01	4	0	0.00	4	0	0.00
3-Insertions_4.col	281	1,046	2.66	2	281	4	0.02	4	0	0.05	4	0	0.02
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	4	0.11	4	0	8.90	4	0	1.36
4-FullIns_3.col	114	541	8.40	6	114	7	0.01	7	0	0.01	7	0	0.01
4-FullIns_4.col	690	6,650	2.80	6	690	8	0.06	8	0	0.57	8	0	0.11
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	9	7.36	9	0	183.76	9	0	25.50

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
4-Insertions_3.col	79	156	5.06	2	79	4	0.02	4	0	0.00	4	0	0.00
4-Insertions_4.col	475	1,795	1.59	2	475	4	0.01	4	0	0.22	4	0	0.05
5-FullIns_3.col	154	792	6.72	7	136	8	0.02	8	0	0.01	8	0	0.01
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	9	0.17	9	0	2.10	9	0	0.48
abb313GPIA.col	1,557	53,356	4.41	8	1,552	14	†	14	709	†	14	0	388.17
anna.col	138	493	5.22	11	19	11	0.02	11	0	0.00	11	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	4	0.02	4	0	0.41	4	0	0.12
ash608GPIA.col	1,216	7,844	1.06	3	1,216	4	0.03	4	0	4.72	4	0	0.91
ash958GPIA.col	1,916	12,506	0.68	3	1,916	4	0.08	4	0	22.63	4	0	3.59
C2000.5.col	2,000	999,836	50.02	16	2,000	15	†	15	1,642	†	16	1,553	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	15	†	15	3,661	†	16	3,542	†
david.col	87	406	10.85	11	22	11	0.01	11	0	0.00	11	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	7	3.17	7	0	3.71	7	0	0.77
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	15	†	15	652	†	16	550	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	33	†	34	924	†	34	923	†
DSJC125.1.col	125	736	9.50	4	125	5	0.01	5	0	0.01	5	0	0.01
DSJC125.5.col	125	3,891	50.21	10	125	13	0.42	13	0	0.45	13	0	0.16
DSJC125.9.col	125	6,961	89.82	34	125	34	†	34	47	†	34	44	†
DSJC250.1.col	250	3,218	10.34	4	250	6	0.02	6	0	0.03	6	0	0.02
DSJC250.5.col	250	15,668	50.34	12	250	14	58.08	14	0	80.25	14	0	13.53
DSJC250.9.col	250	27,897	89.63	43	250	35	†	36	161	†	36	159	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
DSJC500.1.col	500	12,458	9.99	5	500	6	0.42	6	0	0.43	6	0	0.11
DSJC500.5.col	500	62,624	50.20	13	500	15	†	15	152	†	16	53	†
DSJC500.9.col	500	112,437	90.13	56	500	34	†	36	413	†	36	412	†
DSJR500.1.col	500	3,555	2.85	11	328	14	0.01	14	0	0.02	14	0	0.02
DSJR500.1c.col	500	121,275	97.21	83	500	57	†	57	420	†	57	420	†
DSJR500.5.col	500	58,862	47.18	122	488	81	†	104	65	†	123	0	499.27
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	14	†	14	653	†	16	524	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	15	†	15	636	†	15	561	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	15	†	15	642	†	16	532	†
flat300_20_0.col	300	21,375	47.66	11	300	14	107.55	14	0	111.39	14	0	23.44
flat300_26_0.col	300	21,633	48.23	11	300	14	97.91	14	0	102.77	14	0	21.74
flat300_28_0.col	300	21,695	48.37	12	300	14	114.83	14	0	119.88	14	0	25.42
fpsol2.i.1.col	496	11,654	9.49	65	85	66	0.02	66	0	0.00	66	0	0.01
fpsol2.i.2.col	451	8,691	8.57	30	165	31	0.02	31	0	0.01	31	0	0.01
fpsol2.i.3.col	425	8,688	9.64	30	164	31	0.02	31	0	0.01	31	0	0.01
games120.col	120	638	8.94	9	120	10	0.01	10	0	0.00	10	0	0.00
homer.col	561	1,628	1.04	13	35	13	0.01	13	0	0.00	13	0	0.00
huck.col	74	301	11.14	11	20	11	0.01	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	122	55	1.92	55	0	0.10	55	0	0.11
inithx.i.2.col	645	13,979	6.73	31	226	31	0.09	31	0	0.04	31	0	0.05
inithx.i.3.col	621	13,969	7.26	31	212	31	0.09	31	0	0.04	31	0	0.04

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
jean.col	80	254	8.04	10	20	10	0.01	10	0	0.00	10	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	774	†	90	771	†
le450_15a.col	450	8,168	8.09	15	414	15	0.05	15	0	0.07	15	0	0.03
le450_15b.col	450	8,169	8.09	15	417	15	0.06	15	0	0.07	15	0	0.03
le450_15c.col	450	16,680	16.51	15	450	15	0.22	15	0	0.24	15	0	0.08
le450_15d.col	450	16,750	16.58	15	450	15	0.25	15	0	0.30	15	0	0.09
le450_25a.col	450	8,260	8.18	25	272	25	0.02	25	0	0.02	25	0	0.01
le450_25b.col	450	8,263	8.18	25	304	25	0.02	25	0	0.03	25	0	0.02
le450_25c.col	450	17,343	17.17	25	436	25	0.13	25	0	0.16	25	0	0.07
le450_25d.col	450	17,425	17.25	25	438	25	0.09	25	0	0.14	25	0	0.05
le450_5a.col	450	5,714	5.66	5	450	6	0.06	6	0	0.14	6	0	0.04
le450_5b.col	450	5,734	5.68	5	450	6	0.06	6	0	0.10	6	0	0.04
le450_5c.col	450	9,803	9.70	5	450	7	0.16	7	0	0.18	7	0	0.05
le450_5d.col	450	9,757	9.66	5	450	7	0.16	7	0	0.18	7	0	0.05
miles1000.col	128	3,216	39.57	42	51	43	0.01	43	0	0.00	43	0	0.00
miles1500.col	128	5,198	63.95	73	84	73	0.13	73	0	0.05	73	0	0.06
miles250.col	128	387	4.76	8	27	9	0.01	9	0	0.00	9	0	0.00
miles500.col	128	1,170	14.40	20	29	21	0.02	21	0	0.00	21	0	0.00
miles750.col	128	2,113	26.00	31	39	33	0.01	33	0	0.00	33	0	0.00
mug100_1.col	100	166	3.35	3	100	4	0.02	4	0	0.00	4	0	0.00
mug100_25.col	100	166	3.35	3	100	4	0.01	4	0	0.00	4	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
mug88_1.col	88	146	3.81	3	88	4	0.01	4	0	0.00	4	0	0.00
mug88_25.col	88	146	3.81	3	88	4	0.01	4	0	0.00	4	0	0.00
mulsol.i.1.col	197	3,925	20.33	49	56	50	0.01	50	0	0.00	50	0	0.00
mulsol.i.2.col	188	3,885	22.10	31	116	31	0.12	31	0	0.07	31	0	0.03
mulsol.i.3.col	184	3,916	23.26	31	117	31	0.08	31	0	0.05	31	0	0.04
mulsol.i.4.col	185	3,946	23.18	31	118	31	0.08	31	0	0.05	31	0	0.03
mulsol.i.5.col	186	3,973	23.09	31	119	31	0.08	31	0	0.07	31	0	0.03
myciel3.col	11	20	36.36	2	11	4	0.01	4	0	0.00	4	0	0.00
myciel4.col	23	71	28.06	2	23	4	0.01	4	0	0.00	4	0	0.00
myciel5.col	47	236	21.83	2	47	4	0.01	4	0	0.00	4	0	0.00
myciel6.col	95	755	16.91	2	95	4	0.01	4	0	0.01	4	0	0.00
myciel7.col	191	2,360	13.01	2	191	4	0.05	4	0	0.05	4	0	0.02
qg.order30.col	900	26,100	6.45	30	900	30	1.47	30	0	1.57	30	0	0.35
qg.order40.col	1,600	62,400	4.88	40	1,600	40	8.27	40	0	14.41	40	0	2.15
qg.order60.col	3,600	212,400	3.28	60	3,600	60	92.63	60	0	210.54	60	0	26.49
queen10_10.col	100	1,470	29.70	10	100	10	0.01	10	0	0.01	10	0	0.01
queen11_11.col	121	1,980	27.27	11	121	11	0.02	11	0	0.02	11	0	0.01
queen12_12.col	144	2,596	25.21	12	144	12	0.03	12	0	0.04	12	0	0.01
queen13_13.col	169	3,328	23.44	13	169	13	0.04	13	0	0.04	13	0	0.02
queen14_14.col	196	4,186	21.91	14	196	14	0.06	14	0	0.06	14	0	0.02
queen15_15.col	225	5,180	20.56	15	225	15	0.08	15	0	0.11	15	0	0.03

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
queen16_16.col	256	6,320	19.36	16	256	16	0.11	16	0	0.16	16	0	0.04
queen5_5.col	25	160	53.33	5	25	6	0.01	6	0	0.00	6	0	0.00
queen6_6.col	36	290	46.03	6	36	6	0.01	6	0	0.00	6	0	0.00
queen7_7.col	49	476	40.48	7	49	7	0.01	7	0	0.00	7	0	0.00
queen8_12.col	96	1,368	30.00	12	96	12	0.01	12	0	0.01	12	0	0.01
queen8_8.col	64	728	36.11	8	64	8	0.01	8	0	0.01	8	0	0.00
queen9_9.col	81	1,056	32.59	9	81	9	0.01	9	0	0.01	9	0	0.00
r1000.1.col	1,000	14,378	2.88	20	463	21	0.01	21	0	0.04	21	0	0.03
r1000.1c.col	1,000	485,090	97.12	91	1,000	59	†	59	919	†	59	918	†
r1000.5.col	1,000	238,267	47.70	234	984	153	†	153	386	†	153	383	†
r125.1.col	125	209	2.70	5	57	6	0.01	6	0	0.00	6	0	0.00
r125.1c.col	125	7,501	96.79	46	125	47	†	48	56	†	48	55	†
r125.5.col	125	3,838	49.52	36	119	36	0.16	36	0	0.10	36	0	0.10
r250.1.col	250	867	2.79	8	70	8	0.01	8	0	0.00	8	0	0.00
r250.1c.col	250	30,227	97.12	63	250	50	†	50	178	†	50	178	†
r250.5.col	250	14,849	47.71	65	237	65	3.23	65	0	1.27	65	0	0.53
school1.col	385	19,095	25.83	14	361	28	†	28	65	†	28	64	†
school1_nsh.col	352	14,612	23.65	14	331	28	196.52	28	0	116.57	28	0	80.55
wap01a.col	2,368	110,871	3.96	41	2,107	41	11.03	41	0	25.35	41	0	5.94
wap02a.col	2,464	111,742	3.68	40	2,248	40	15.60	40	0	46.50	40	0	8.88
wap03a.col	4,730	286,722	2.56	40	4,701	41	†	41	98	†	41	93	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
wap04a.col	5,231	294,902	2.16	40	5,204	41	142.63	40	184	†	41	0	99.33
wap05a.col	905	43,081	10.53	50	675	50	0.44	50	0	0.44	50	0	0.23
wap06a.col	947	43,571	9.73	40	807	40	3.32	40	0	3.08	40	0	1.27
wap07a.col	1,809	103,368	6.32	40	1,701	41	14.59	41	0	23.83	41	0	5.78
wap08a.col	1,870	104,176	5.96	40	1,753	40	15.12	40	0	24.48	40	0	5.37
will199GPIA.col	701	6,772	2.76	6	700	8	0.05	8	0	0.53	8	0	0.14
zeroin.i.1.col	211	4,100	18.51	49	73	49	0.11	49	0	0.04	49	0	0.06
zeroin.i.2.col	211	3,541	15.98	30	106	30	0.03	30	0	0.01	30	0	0.01
zeroin.i.3.col	206	3,540	16.77	30	106	30	0.02	30	0	0.01	30	0	0.01

Table A.5: RDS performance for 3-plex.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	24	†	24	110	†	25	104	†
brock200_2.clq	200	9,876	49.63	12	200	15	†	15	25	†	16	0	433.99
brock200_3.clq	200	12,048	60.54	15	200	18	†	18	76	†	18	60	†
brock200_4.clq	200	13,089	65.77	17	200	20	†	20	85	†	20	78	†
brock400_1.clq	400	59,723	74.84	27	400	25	†	25	305	†	25	303	†
brock400_2.clq	400	59,786	74.92	29	400	23	†	23	314	†	24	309	†
brock400_3.clq	400	59,681	74.79	31	400	24	†	25	312	†	25	305	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock400_4.clq	400	59,765	74.89	33	400	22	†	22	319	†	23	311	†
brock800_1.clq	800	207,505	64.93	23	800	19	†	19	686	†	20	672	†
brock800_2.clq	800	208,166	65.13	24	800	20	†	20	688	†	21	677	†
brock800_3.clq	800	207,333	64.87	25	800	20	†	20	684	†	20	675	†
brock800_4.clq	800	207,643	64.97	26	800	20	†	20	684	†	20	673	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.02	12	0	0.06	12	0	0.03
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.01	24	0	0.02	24	0	0.02
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.02	58	0	0.01	58	0	0.01
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.01	14	0	1.46	14	0	0.32
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.05	126	0	0.04	126	0	0.06
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.01	26	0	0.33	26	0	0.11
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.02	64	0	0.07	64	0	0.06
hamming10-2.clq	1,024	518,656	99.02	512	1,024	100	†	135	757	†	135	757	†
hamming10-4.clq	1,024	434,176	82.89	40	1,024	16	†	17	858	†	18	832	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.17	32	0	0.09	32	0	0.08
hamming6-4.clq	64	704	34.92	4	64	8	0.02	8	0	0.01	8	0	0.01
hamming8-2.clq	256	31,616	96.86	128	256	100	†	128	0	196.09	128	0	183.06
hamming8-4.clq	256	20,864	63.92	16	256	16	†	16	94	†	18	64	†
johnson16-2-4.clq	120	5,460	76.47	8	120	15	†	15	35	†	16	32	†
johnson32-2-4.clq	496	107,880	87.88	16	496	24	†	24	275	†	24	275	†
johnson8-2-4.clq	28	210	55.56	4	28	8	0.02	8	0	0.00	8	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
johnson8-4-4.clq	70	1,855	76.81	14	70	18	5.32	18	0	4.64	18	0	2.58
keller4.clq	171	9,435	64.91	11	171	21	†	21	7	†	21	3	†
keller5.clq	776	225,990	75.16	27	776	22	†	22	608	†	22	605	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	22	†	22	3,193	†	22	3,190	†
MANN_a27.clq	378	70,551	99.02	126	378	351	†	351	21	†	351	21	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	990	†	990	41	†	990	41	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	1,842	†	3,240	78	†	3,240	78	†
MANN_a9.clq	45	918	92.73	16	45	36	0.31	36	0	0.15	36	0	0.14
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	13	†	13	607	†	14	480	†
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	27	†	29	829	†	31	816	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	30	†	34	899	†	34	896	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	14	†	14	1,052	†	14	985	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	29	†	29	1,350	†	29	1,342	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	33	†	36	1,404	†	36	1,403	†
p_hat300-1.clq	300	10,933	24.38	8	300	12	73.81	12	0	77.87	12	0	18.53
p_hat300-2.clq	300	21,928	48.89	25	300	27	†	27	148	†	28	141	†
p_hat300-3.clq	300	33,390	74.45	36	300	32	†	33	199	†	33	198	†
p_hat500-1.clq	500	31,569	25.31	9	500	13	†	13	101	†	14	0	473.33
p_hat500-2.clq	500	62,946	50.46	36	500	31	†	32	332	†	32	322	†
p_hat500-3.clq	500	93,800	75.19	50	500	35	†	36	402	†	37	396	†
p_hat700-1.clq	700	60,999	24.93	11	700	13	†	13	291	†	14	158	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat700-2.clq	700	121,728	49.76	44	700	30	†	31	510	†	32	495	†
p_hat700-3.clq	700	183,010	74.81	62	700	29	†	29	603	†	31	598	†
san1000.clq	1,000	250,500	50.15	15	1,000	24	†	24	912	†	24	911	†
san200_0.7_1.clq	200	13,930	70.00	30	200	41	†	41	100	†	41	100	†
san200_0.7_2.clq	200	13,930	70.00	18	200	34	†	34	121	†	34	120	†
san200_0.9_1.clq	200	17,910	90.00	70	200	125	159.25	125	0	77.96	125	0	47.48
san200_0.9_2.clq	200	17,910	90.00	60	200	47	†	50	112	†	50	111	†
san200_0.9_3.clq	200	17,910	90.00	44	200	36	†	36	133	†	36	133	†
san400_0.5_1.clq	400	39,900	50.00	13	400	20	†	20	319	†	20	318	†
san400_0.7_1.clq	400	55,860	70.00	40	400	48	†	49	278	†	50	276	†
san400_0.7_2.clq	400	55,860	70.00	30	400	41	†	41	294	†	41	294	†
san400_0.7_3.clq	400	55,860	70.00	22	400	33	†	33	307	†	33	307	†
san400_0.9_1.clq	400	71,820	90.00	100	400	49	†	62	285	†	63	282	†
sanr200_0.7.clq	200	13,868	69.69	18	200	21	†	21	99	†	22	92	†
sanr200_0.9.clq	200	17,863	89.76	42	200	37	†	38	129	†	39	128	†
sanr400_0.5.clq	400	39,984	50.11	13	400	16	†	16	227	†	16	200	†
sanr400_0.7.clq	400	55,869	70.01	21	400	21	†	21	308	†	22	297	†
adjnoun.graph	112	425	6.84	5	102	8	0.02	8	0	0.03	8	0	0.01
as-22july06.graph	22,963	48,436	0.02	17	182	21	0.20	21	0	0.16	21	0	0.10
astro-ph.graph	16,706	121,251	0.09	57	113	57	0.01	57	0	0.01	57	0	0.01
caidaRouterLevel.graph	192,244	609,066	0.00	17	4,704	23	249.67	8	3,479	†	12	2,476	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
celegans_metabolic.graph	453	2,025	1.98	9	138	11	0.05	11	0	0.06	11	0	0.02
celegansneural.graph	297	2,148	4.89	8	265	11	0.20	11	0	0.54	11	0	0.15
chesapeake.graph	39	170	22.94	5	39	8	0.02	8	0	0.00	8	0	0.00
cnr-2000.graph	325,557	2,738,969	0.01	84	89	86	0.01	86	0	0.00	86	0	0.01
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.02	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.01	115	0	0.00	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	53	18	0.01	18	0	0.00	18	0	0.00
cond-mat-2003.graph	31,163	120,029	0.03	25	50	26	0.02	26	0	0.00	26	0	0.00
cond-mat-2005.graph	40,421	175,691	0.02	30	30	30	0.01	30	0	0.00	30	0	0.00
dolphins.graph	62	159	8.41	5	53	7	0.01	7	0	0.00	7	0	0.00
email.graph	1,133	5,451	0.85	12	238	12	0.19	12	0	1.37	12	0	0.21
football.graph	115	613	9.35	9	115	11	0.02	11	0	0.03	11	0	0.01
hep-th.graph	8,361	15,751	0.05	24	24	24	0.01	24	0	0.00	24	0	0.01
jazz.graph	198	2,742	14.06	30	30	30	0.01	30	0	0.00	30	0	0.05
karate.graph	34	78	13.90	5	33	6	0.02	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	31	12	0.01	12	0	0.00	12	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.02	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	145	31	0.03	31	0	0.03	31	0	0.03
polblogs.graph	1,490	16,715	1.51	20	489	27	16.15	27	0	18.82	27	0	6.51
polbooks.graph	105	441	8.08	6	103	9	0.01	9	0	0.03	9	0	0.01

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
power.graph	4,941	6,594	0.05	6	231	6	0.01	6	0	1.21	6	0	0.20
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	650	17	0.01	17	0	27.01	17	0	3.81
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	211	19	0.02	19	0	0.48	19	0	0.10
Cit-HepTh.txt	27,769	352,285	0.09	23	7,744	31	†	13	6,628	†	22	5,117	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,051	22	86.19	8	895	†	22	0	141.13
Slashdot0811.txt	77,360	469,180	0.02	26	6,571	8	†	6	5,520	†	6	5,001	†
Slashdot0902.txt	82,168	504,230	0.02	27	6,532	8	†	6	5,498	†	7	4,796	†
soc-Epinions1.txt	75,879	405,740	0.01	23	5,456	27	†	13	4,477	†	27	3,253	†
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	2.09	202	0	2.31	202	0	0.42
web-Google.txt	875,713	4,322,051	0.00	44	222	45	†	47	0	1.66	47	0	0.72
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	152	†	152	241	†	155	0	270.76
web-Stanford.txt	281,903	1,992,636	0.01	61	1,439	59	†	43	630	†	43	591	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,452	23	†	18	1,141	†	24	120	†
1-FullIns_3.col	30	100	22.99	3	30	7	0.01	7	0	0.00	7	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	7	0.02	7	0	0.03	7	0	0.02
1-FullIns_5.col	282	3,247	8.20	3	282	8	0.59	8	0	1.69	8	0	0.49
1-Insertions_4.col	67	232	10.49	2	67	5	0.02	5	0	0.01	5	0	0.01
1-Insertions_5.col	202	1,227	6.04	2	202	6	0.05	6	0	0.47	6	0	0.13
1-Insertions_6.col	607	6,337	3.45	2	607	6	2.23	6	0	34.81	6	0	5.38
2-FullIns_3.col	52	201	15.16	4	52	7	0.02	7	0	0.00	7	0	0.00
2-FullIns_4.col	212	1,621	7.25	4	212	8	0.06	8	0	0.47	8	0	0.15

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
2-FullIns_5.col	852	12,201	3.37	4	852	8	7.85	8	0	96.35	8	0	17.72
2-Insertions_3.col	37	72	10.81	2	37	4	0.01	4	0	0.00	4	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	5	0.02	5	0	0.20	5	0	0.08
2-Insertions_5.col	597	3,936	2.21	2	597	6	0.41	6	0	33.61	6	0	5.20
3-FullIns_3.col	80	346	10.95	5	80	7	0.02	7	0	0.02	7	0	0.01
3-FullIns_4.col	405	3,524	4.31	5	405	9	0.34	9	0	5.59	9	0	1.10
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	10	92.06	6	973	†	6	468	†
3-Insertions_3.col	56	110	7.14	2	56	4	0.02	4	0	0.01	4	0	0.00
3-Insertions_4.col	281	1,046	2.66	2	281	5	0.01	5	0	2.47	5	0	0.40
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	6	3.76	6	367	†	6	0	355.80
4-FullIns_3.col	114	541	8.40	6	114	8	0.02	8	0	0.08	8	0	0.02
4-FullIns_4.col	690	6,650	2.80	6	690	10	1.37	10	0	53.06	10	0	7.01
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	6	†	6	3,132	†	6	2,610	†
4-Insertions_3.col	79	156	5.06	2	79	4	0.03	4	0	0.02	4	0	0.01
4-Insertions_4.col	475	1,795	1.59	2	475	5	0.03	5	0	19.49	5	0	2.78
5-FullIns_3.col	154	792	6.72	7	154	9	0.02	9	0	0.19	9	0	0.10
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	11	4.12	11	0	396.96	11	0	69.92
abb313GPIA.col	1,557	53,356	4.41	8	1,552	17	†	17	1,176	†	17	1,165	†
anna.col	138	493	5.22	11	19	11	0.02	11	0	0.00	11	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	6	0.23	6	0	65.57	6	0	10.34
ash608GPIA.col	1,216	7,844	1.06	3	1,216	6	0.45	6	165	†	6	0	205.45

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
ash958GPIA.col	1,916	12,506	0.68	3	1,916	6	0.73	6	877	†	6	358	†
C2000.5.col	2,000	999,836	50.02	16	2,000	15	†	15	1,819	†	16	1,780	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	15	†	15	3,820	†	16	3,778	†
david.col	87	406	10.85	11	33	11	0.01	11	0	0.00	11	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	8	†	8	139	†	8	0	189.29
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	16	†	16	815	†	16	790	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	36	†	37	935	†	37	934	†
DSJC125.1.col	125	736	9.50	4	125	7	0.03	7	0	0.09	7	0	0.03
DSJC125.5.col	125	3,891	50.21	10	125	14	44.15	14	0	47.42	14	0	14.25
DSJC125.9.col	125	6,961	89.82	34	125	37	†	39	55	†	39	55	†
DSJC250.1.col	250	3,218	10.34	4	250	7	1.70	7	0	1.99	7	0	0.43
DSJC250.5.col	250	15,668	50.34	12	250	15	†	15	85	†	16	45	†
DSJC250.9.col	250	27,897	89.63	43	250	35	†	37	180	†	37	180	†
DSJC500.1.col	500	12,458	9.99	5	500	8	37.75	8	0	42.44	8	0	6.37
DSJC500.5.col	500	62,624	50.20	13	500	16	†	16	319	†	16	289	†
DSJC500.9.col	500	112,437	90.13	56	500	37	†	37	431	†	37	430	†
DSJR500.1.col	500	3,555	2.85	11	423	15	0.02	15	0	1.53	15	0	0.27
DSJR500.1c.col	500	121,275	97.21	83	500	71	†	75	410	†	75	410	†
DSJR500.5.col	500	58,862	47.18	122	489	73	†	78	186	†	79	184	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	15	†	15	810	†	16	773	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	15	†	15	820	†	16	782	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	15	†	15	819	†	16	780	†
flat300_20_0.col	300	21,375	47.66	11	300	15	†	15	109	†	15	75	†
flat300_26_0.col	300	21,633	48.23	11	300	15	†	15	110	†	16	67	†
flat300_28_0.col	300	21,695	48.37	12	300	15	†	15	110	†	15	80	†
fpsol2.i.1.col	496	11,654	9.49	65	86	66	0.30	66	0	0.08	66	0	0.11
fpsol2.i.2.col	451	8,691	8.57	30	203	31	0.34	31	0	0.18	31	0	0.12
fpsol2.i.3.col	425	8,688	9.64	30	203	31	0.36	31	0	0.18	31	0	0.13
games120.col	120	638	8.94	9	120	10	0.02	10	0	0.02	10	0	0.01
homer.col	561	1,628	1.04	13	61	13	0.02	13	0	0.00	13	0	0.00
huck.col	74	301	11.14	11	32	11	0.02	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	143	56	6.83	56	0	1.71	56	0	1.47
inithx.i.2.col	645	13,979	6.73	31	278	32	2.39	32	0	1.38	32	0	1.34
inithx.i.3.col	621	13,969	7.26	31	268	32	2.08	32	0	1.16	32	0	1.14
jean.col	80	254	8.04	10	31	12	0.01	12	0	0.00	12	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	795	†	90	794	†
le450_15a.col	450	8,168	8.09	15	419	15	1.48	15	0	2.50	15	0	0.40
le450_15b.col	450	8,169	8.09	15	421	15	2.87	15	0	4.54	15	0	0.71
le450_15c.col	450	16,680	16.51	15	450	15	17.77	15	0	21.76	15	0	3.74
le450_15d.col	450	16,750	16.58	15	450	15	23.62	15	0	29.65	15	0	5.38
le450_25a.col	450	8,260	8.18	25	280	25	0.19	25	0	0.23	25	0	0.06
le450_25b.col	450	8,263	8.18	25	308	25	0.36	25	0	0.48	25	0	0.11

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
le450_25c.col	450	17,343	17.17	25	438	25	8.11	25	0	9.70	25	0	1.96
le450_25d.col	450	17,425	17.25	25	440	25	3.95	25	0	4.87	25	0	1.17
le450_5a.col	450	5,714	5.66	5	450	8	2.56	8	0	11.71	8	0	1.50
le450_5b.col	450	5,734	5.68	5	450	8	3.23	8	0	14.33	8	0	1.93
le450_5c.col	450	9,803	9.70	5	450	9	16.12	9	0	21.55	9	0	3.19
le450_5d.col	450	9,757	9.66	5	450	9	13.65	9	0	17.94	9	0	2.66
miles1000.col	128	3,216	39.57	42	61	44	0.01	44	0	0.01	44	0	0.01
miles1500.col	128	5,198	63.95	73	85	73	5.99	73	0	1.84	73	0	1.56
miles250.col	128	387	4.76	8	41	10	0.01	10	0	0.00	10	0	0.00
miles500.col	128	1,170	14.40	20	35	22	0.01	22	0	0.00	22	0	0.00
miles750.col	128	2,113	26.00	31	41	33	0.02	33	0	0.00	33	0	0.00
mug100_1.col	100	166	3.35	3	100	5	0.02	5	0	0.04	5	0	0.02
mug100_25.col	100	166	3.35	3	100	5	0.01	5	0	0.04	5	0	0.01
mug88_1.col	88	146	3.81	3	88	5	0.02	5	0	0.02	5	0	0.01
mug88_25.col	88	146	3.81	3	88	5	0.01	5	0	0.02	5	0	0.01
mulsol.i.1.col	197	3,925	20.33	49	57	51	0.02	51	0	0.01	51	0	0.01
mulsol.i.2.col	188	3,885	22.10	31	119	32	0.11	32	0	0.05	32	0	0.04
mulsol.i.3.col	184	3,916	23.26	31	120	32	0.13	32	0	0.06	32	0	0.06
mulsol.i.4.col	185	3,946	23.18	31	121	32	0.11	32	0	0.07	32	0	0.06
mulsol.i.5.col	186	3,973	23.09	31	122	32	0.14	32	0	0.08	32	0	0.06
myciel3.col	11	20	36.36	2	11	5	0.01	5	0	0.00	5	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
myciel4.col	23	71	28.06	2	23	5	0.01	5	0	0.00	5	0	0.00
myciel5.col	47	236	21.83	2	47	6	0.02	6	0	0.01	6	0	0.00
myciel6.col	95	755	16.91	2	95	6	0.08	6	0	0.08	6	0	0.03
myciel7.col	191	2,360	13.01	2	191	6	1.33	6	0	1.37	6	0	0.42
qg.order30.col	900	26,100	6.45	30	900	30	339.30	30	0	371.78	30	0	51.65
qg.order40.col	1,600	62,400	4.88	40	1,600	40	†	40	610	†	40	60	†
qg.order60.col	3,600	212,400	3.28	60	3,600	60	†	60	2,583	†	60	2,073	†
queen10_10.col	100	1,470	29.70	10	100	10	0.39	10	0	0.41	10	0	0.11
queen11_11.col	121	1,980	27.27	11	121	11	0.70	11	0	0.73	11	0	0.19
queen12_12.col	144	2,596	25.21	12	144	12	1.22	12	0	1.30	12	0	0.35
queen13_13.col	169	3,328	23.44	13	169	13	2.12	13	0	2.21	13	0	0.48
queen14_14.col	196	4,186	21.91	14	196	14	3.43	14	0	3.41	14	0	0.69
queen15_15.col	225	5,180	20.56	15	225	15	5.43	15	0	5.47	15	0	1.06
queen16_16.col	256	6,320	19.36	16	256	16	8.41	16	0	8.63	16	0	1.79
queen5_5.col	25	160	53.33	5	25	9	0.01	9	0	0.00	9	0	0.00
queen6_6.col	36	290	46.03	6	36	9	0.02	9	0	0.01	9	0	0.01
queen7_7.col	49	476	40.48	7	49	9	0.03	9	0	0.03	9	0	0.02
queen8_12.col	96	1,368	30.00	12	96	12	0.23	12	0	0.25	12	0	0.07
queen8_8.col	64	728	36.11	8	64	9	0.09	9	0	0.09	9	0	0.04
queen9_9.col	81	1,056	32.59	9	81	9	0.22	9	0	0.21	9	0	0.07
r1000.1.col	1,000	14,378	2.88	20	665	22	0.08	22	0	8.32	22	0	1.16

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
r1000.1c.col	1,000	485,090	97.12	91	1,000	65	†	65	920	†	65	920	†
r1000.5.col	1,000	238,267	47.70	234	984	82	†	119	541	†	134	480	†
r125.1.col	125	209	2.70	5	101	6	0.01	6	0	0.04	6	0	0.01
r125.1c.col	125	7,501	96.79	46	125	62	†	62	50	†	62	49	†
r125.5.col	125	3,838	49.52	36	119	38	2.78	38	0	1.63	38	0	1.32
r250.1.col	250	867	2.79	8	140	9	0.01	9	0	0.12	9	0	0.03
r250.1c.col	250	30,227	97.12	63	250	65	†	66	170	†	67	169	†
r250.5.col	250	14,849	47.71	65	237	67	165.28	67	0	51.06	67	0	35.05
school1.col	385	19,095	25.83	14	363	34	†	34	115	†	35	67	†
school1_nsh.col	352	14,612	23.65	14	332	37	42.51	37	0	31.08	37	0	16.49
wap01a.col	2,368	110,871	3.96	41	2,166	41	†	41	983	†	41	439	†
wap02a.col	2,464	111,742	3.68	40	2,362	41	†	41	1,120	†	41	576	†
wap03a.col	4,730	286,722	2.56	40	4,702	41	†	40	3,491	†	41	2,814	†
wap04a.col	5,231	294,902	2.16	40	5,205	40	†	40	3,703	†	40	3,333	†
wap05a.col	905	43,081	10.53	50	679	50	21.39	50	0	23.61	50	0	6.41
wap06a.col	947	43,571	9.73	40	834	40	199.32	40	0	276.29	40	0	55.84
wap07a.col	1,809	103,368	6.32	40	1,710	42	†	42	596	†	42	79	†
wap08a.col	1,870	104,176	5.96	40	1,763	40	†	40	728	†	40	230	†
will199GPIA.col	701	6,772	2.76	6	700	10	0.66	10	0	98.09	10	0	14.68
zeroin.i.1.col	211	4,100	18.51	49	79	50	1.79	50	0	0.83	50	0	0.89
zeroin.i.2.col	211	3,541	15.98	30	131	32	0.33	32	0	0.22	32	0	0.23

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
zeroin.i.3.col	206	3,540	16.77	30	131	32	0.31	32	0	0.22	32	0	0.23

Table A.6: RDS performance for 4-plex.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	26	600	26	127	†	26	125	†
brock200_2.clq	200	9,876	49.63	12	200	17	600	16	88	†	17	67	†
brock200_3.clq	200	12,048	60.54	15	200	18	600	18	119	†	19	107	†
brock200_4.clq	200	13,089	65.77	17	200	21	600	21	122	†	21	113	†
brock400_1.clq	400	59,723	74.84	27	400	26	600	26	328	†	26	327	†
brock400_2.clq	400	59,786	74.92	29	400	23	600	23	333	†	24	329	†
brock400_3.clq	400	59,681	74.79	31	400	26	600	26	332	†	26	329	†
brock400_4.clq	400	59,765	74.89	33	400	24	600	24	334	†	25	329	†
brock800_1.clq	800	207,505	64.93	23	800	21	600	21	716	†	21	710	†
brock800_2.clq	800	208,166	65.13	24	800	21	600	21	720	†	22	715	†
brock800_3.clq	800	207,333	64.87	25	800	21	600	21	718	†	22	712	†
brock800_4.clq	800	207,643	64.97	26	800	21	600	21	716	†	21	710	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.01	12	0	3.24	12	0	0.74
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.02	24	0	0.44	24	0	0.18
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.03	58	0	0.07	58	0	0.07

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.03	14	0	239.12	14	0	38.59
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.22	126	0	0.40	126	0	0.33
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.03	26	0	18.33	26	0	3.63
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.08	64	0	1.37	64	0	0.52
hamming10-2.clq	1,024	518,656	99.02	512	1,024	44	600	45	952	†	45	952	†
hamming10-4.clq	1,024	434,176	82.89	40	1,024	18	600	18	889	†	18	887	†
hamming6-2.clq	64	1,824	90.48	32	64	40	225.13	40	0	94.25	40	0	72.27
hamming6-4.clq	64	704	34.92	4	64	10	0.13	10	0	0.12	10	0	0.05
hamming8-2.clq	256	31,616	96.86	128	256	44	600	45	184	†	45	184	†
hamming8-4.clq	256	20,864	63.92	16	256	18	600	18	121	†	18	119	†
johnson16-2-4.clq	120	5,460	76.47	8	120	18	600	18	50	†	18	49	†
johnson32-2-4.clq	496	107,880	87.88	16	496	25	600	25	323	†	25	323	†
johnson8-2-4.clq	28	210	55.56	4	28	9	0.03	9	0	0.05	9	0	0.02
johnson8-4-4.clq	70	1,855	76.81	14	70	21	600	21	5	†	22	0	311.86
keller4.clq	171	9,435	64.91	11	171	16	600	16	72	†	23	32	†
keller5.clq	776	225,990	75.16	27	776	16	600	16	677	†	24	631	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	16	600	16	3,262	†	24	3,216	†
MANN_a27.clq	378	70,551	99.02	126	378	351	600	351	21	†	351	21	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	990	600	990	41	†	990	41	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	1,843	600	3,240	78	†	3,240	78	†
MANN_a9.clq	45	918	92.73	16	45	36	26.27	36	0	12.62	36	0	9.47

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	14	600	14	790	†	14	752	†
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	26	600	26	889	†	26	880	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	33	600	33	921	†	33	920	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	13	600	13	1,300	†	14	1,242	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	31	600	31	1,389	†	32	1,385	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	34	600	34	1,423	†	34	1,422	†
p_hat300-1.clq	300	10,933	24.38	8	300	13	600	13	90	†	14	29	†
p_hat300-2.clq	300	21,928	48.89	25	300	27	600	28	184	†	29	178	†
p_hat300-3.clq	300	33,390	74.45	36	300	31	600	32	222	†	32	221	†
p_hat500-1.clq	500	31,569	25.31	9	500	14	600	14	276	†	14	237	†
p_hat500-2.clq	500	62,946	50.46	36	500	29	600	29	382	†	30	375	†
p_hat500-3.clq	500	93,800	75.19	50	500	33	600	35	421	†	35	418	†
p_hat700-1.clq	700	60,999	24.93	11	700	13	600	13	483	†	14	416	†
p_hat700-2.clq	700	121,728	49.76	44	700	29	600	29	562	†	30	558	†
p_hat700-3.clq	700	183,010	74.81	62	700	29	600	29	622	†	30	620	†
san1000.clq	1,000	250,500	50.15	15	1,000	31	600	31	910	†	31	910	†
san200_0.7_1.clq	200	13,930	70.00	30	200	52	600	53	81	†	53	81	†
san200_0.7_2.clq	200	13,930	70.00	18	200	46	600	46	111	†	46	111	†
san200_0.9_1.clq	200	17,910	90.00	70	200	39	600	39	138	†	39	137	†
san200_0.9_2.clq	200	17,910	90.00	60	200	43	600	44	133	†	44	132	†
san200_0.9_3.clq	200	17,910	90.00	44	200	38	600	38	140	†	39	139	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
san400_0.5_1.clq	400	39,900	50.00	13	400	26	600	26	320	†	26	319	†
san400_0.7_1.clq	400	55,860	70.00	40	400	70	600	70	242	†	70	242	†
san400_0.7_2.clq	400	55,860	70.00	30	400	51	600	51	285	†	51	283	†
san400_0.7_3.clq	400	55,860	70.00	22	400	44	600	44	292	†	44	291	†
san400_0.9_1.clq	400	71,820	90.00	100	400	36	600	38	338	†	38	338	†
sanr200_0.7.clq	200	13,868	69.69	18	200	23	600	23	124	†	23	119	†
sanr200_0.9.clq	200	17,863	89.76	42	200	40	600	40	138	†	41	136	†
sanr400_0.5.clq	400	39,984	50.11	13	400	18	600	18	280	†	18	268	†
sanr400_0.7.clq	400	55,869	70.01	21	400	23	600	23	330	†	23	328	†
adjnoun.graph	112	425	6.84	5	112	8	0.09	8	0	0.72	8	0	0.21
as-22july06.graph	22,963	48,436	0.02	17	204	22	7.47	22	0	5.25	22	0	1.81
astro-ph.graph	16,706	121,251	0.09	57	165	57	0.48	57	0	0.71	57	0	0.23
caidaRouterLevel.graph	192,244	609,066	0.00	17	5,417	11	600	8	4,917	†	9	4,692	†
celegans_metabolic.graph	453	2,025	1.98	9	240	13	1.45	13	0	6.70	13	0	1.43
celegansneural.graph	297	2,148	4.89	8	274	12	4.24	12	0	27.20	12	0	6.71
chesapeake.graph	39	170	22.94	5	39	9	0.02	9	0	0.01	9	0	0.01
cnr-2000.graph	325,557	2,738,969	0.01	84	170	86	0.01	86	0	2.95	86	0	0.51
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.01	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.02	115	0	0.00	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	98	19	0.02	19	0	0.06	19	0	0.02
cond-mat-2003.graph	31,163	120,029	0.03	25	77	27	0.03	27	0	0.08	27	0	0.03

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
cond-mat-2005.graph	40,421	175,691	0.02	30	57	30	0.01	30	0	0.01	30	0	0.01
dolphins.graph	62	159	8.41	5	62	7	0.02	7	0	0.12	7	0	0.03
email.graph	1,133	5,451	0.85	12	349	12	3.7	12	0	308.48	12	0	43.24
football.graph	115	613	9.35	9	115	12	0.02	12	0	0.30	12	0	0.08
hep-th.graph	8,361	15,751	0.05	24	24	24	0.02	24	0	0.00	24	0	0.00
jazz.graph	198	2,742	14.06	30	30	30	0.01	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	34	8	0.02	8	0	0.01	8	0	0.00
lesmis.graph	77	254	8.68	10	38	12	0.01	12	0	0.01	12	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.01	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	171	33	0.11	33	0	0.30	33	0	0.11
polblogs.graph	1,490	16,715	1.51	20	517	29	523.48	20	119	†	29	0	212.94
polbooks.graph	105	441	8.08	6	105	10	0.02	10	0	0.87	10	0	0.17
power.graph	4,941	6,594	0.05	6	3,353	8	0.06	6	2,984	†	8	2,716	†
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	2,016	18	0.03	16	1,654	†	16	1,455	†
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	534	20	0.01	19	166	†	20	0	276.01
Cit-HepTh.txt	27,769	352,285	0.09	23	8,168	33	600	12	7,670	†	12	7,505	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,252	12	600	7	1,849	†	7	1,676	†
Slashdot0811.txt	77,360	469,180	0.02	26	6,974	7	600	5	6,605	†	7	6,293	†
Slashdot0902.txt	82,168	504,230	0.02	27	6,978	9	600	6	6,580	†	8	6,279	†
soc-Epinions1.txt	75,879	405,740	0.01	23	5,719	21	600	17	5,299	†	17	5,119	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	99.87	202	0	170.71	202	0	22.53
web-Google.txt	875,713	4,322,051	0.00	44	223	46	600	48	0	10.01	48	0	3.15
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	150	600	150	932	†	152	720	†
web-Stanford.txt	281,903	1,992,636	0.01	61	1,499	5	600	41	774	†	41	750	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,520	13	600	16	1,970	†	18	1,819	†
1-FullIns_3.col	30	100	22.99	3	30	8	0.02	8	0	0.00	8	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	9	0.06	9	0	0.65	9	0	0.19
1-FullIns_5.col	282	3,247	8.20	3	282	10	6.47	10	0	94.15	10	0	16.67
1-Insertions_4.col	67	232	10.49	2	67	6	0.22	6	0	0.19	6	0	0.06
1-Insertions_5.col	202	1,227	6.04	2	202	8	0.28	8	0	13.40	8	0	2.41
1-Insertions_6.col	607	6,337	3.45	2	607	8	52.92	8	130	†	8	0	452.59
2-FullIns_3.col	52	201	15.16	4	52	8	0.02	8	0	0.05	8	0	0.02
2-FullIns_4.col	212	1,621	7.25	4	212	10	0.64	10	0	29.75	10	0	5.74
2-FullIns_5.col	852	12,201	3.37	4	852	10	258.96	8	425	†	8	263	†
2-Insertions_3.col	37	72	10.81	2	37	6	0	6	0	0.01	6	0	0.01
2-Insertions_4.col	149	541	4.91	2	149	6	8.14	6	0	6.48	6	0	1.38
2-Insertions_5.col	597	3,936	2.21	2	597	8	9.19	8	134	†	8	0	437.03
3-FullIns_3.col	80	346	10.95	5	80	8	0.02	8	0	0.39	8	0	0.11
3-FullIns_4.col	405	3,524	4.31	5	405	10	4.42	8	22	†	10	0	110.23
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	6	600	6	1,638	†	8	1,361	†
3-Insertions_3.col	56	110	7.14	2	56	6	0.06	6	0	0.07	6	0	0.03

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
3-Insertions_4.col	281	1,046	2.66	2	281	6	171.91	6	0	160.35	6	0	22.58
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	8	193.32	7	974	†	7	799	†
4-FullIns_3.col	114	541	8.40	6	114	9	0.03	9	0	1.45	9	0	0.35
4-FullIns_4.col	690	6,650	2.80	6	690	12	22.92	8	320	†	8	125	†
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	6	600	6	3,767	†	6	3,582	†
4-Insertions_3.col	79	156	5.06	2	79	6	0.39	6	0	0.31	6	0	0.10
4-Insertions_4.col	475	1,795	1.59	2	475	6	600	6	107	†	6	0	262.81
5-FullIns_3.col	154	792	6.72	7	154	10	0.06	10	0	5.81	10	0	1.16
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	13	108.76	8	728	†	8	537	†
abb313GPIA.col	1,557	53,356	4.41	8	1,555	21	600	21	1,488	†	22	1,483	†
anna.col	138	493	5.22	11	24	12	0.01	12	0	0.00	12	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	8	2.01	8	283	†	8	98	†
ash608GPIA.col	1,216	7,844	1.06	3	1,216	8	2.95	8	840	†	8	657	†
ash958GPIA.col	1,916	12,506	0.68	3	1,916	8	5.8	8	1,551	†	8	1,377	†
C2000.5.col	2,000	999,836	50.02	16	2,000	17	600	16	1,887	†	17	1,861	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	16		16	3,888	†	16	3,874	†
david.col	87	406	10.85	11	36	13	0.02	13	0	0.00	13	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	8	600	8	675	†	9	507	†
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	16	600	16	892	†	17	875	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	39	600	39	940	†	40	939	†
DSJC125.1.col	125	736	9.50	4	125	8	0.31	8	0	2.71	8	0	0.62

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
DSJC125.5.col	125	3,891	50.21	10	125	16	600	16	15	†	17	0	507.92
DSJC125.9.col	125	6,961	89.82	34	125	39	600	40	64	†	40	64	†
DSJC250.1.col	250	3,218	10.34	4	250	8	99.5	8	0	146.59	8	0	27.21
DSJC250.5.col	250	15,668	50.34	12	250	16	600	16	150	†	17	128	†
DSJC250.9.col	250	27,897	89.63	43	250	36	600	36	193	†	37	191	†
DSJC500.1.col	500	12,458	9.99	5	500	9	600	9	154	†	9	13	†
DSJC500.5.col	500	62,624	50.20	13	500	16	600	16	396	†	17	374	†
DSJC500.9.col	500	112,437	90.13	56	500	42	600	42	433	†	42	433	†
DSJR500.1.col	500	3,555	2.85	11	441	15	0.03	15	0	162.20	15	0	21.47
DSJR500.1c.col	500	121,275	97.21	83	500	84	600	97	392	†	97	392	†
DSJR500.5.col	500	58,862	47.18	122	492	61	600	66	243	†	80	204	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	16	600	16	881	†	16	868	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	16	600	16	890	†	17	869	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	16	600	16	882	†	16	867	†
flat300_20_0.col	300	21,375	47.66	11	300	16	600	16	183	†	16	166	†
flat300_26_0.col	300	21,633	48.23	11	300	16	600	16	182	†	17	158	†
flat300_28_0.col	300	21,695	48.37	12	300	17	600	17	177	†	17	160	†
fpsol2.i.1.col	496	11,654	9.49	65	91	66	16.33	66	0	5.05	66	0	4.82
fpsol2.i.2.col	451	8,691	8.57	30	238	32	17.85	32	0	11.22	32	0	5.10
fpsol2.i.3.col	425	8,688	9.64	30	238	32	17.89	32	0	11.22	32	0	4.92
games120.col	120	638	8.94	9	120	10	0.01	10	0	0.48	10	0	0.14

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
homer.col	561	1,628	1.04	13	68	14	0.03	14	0	0.07	14	0	0.03
huck.col	74	301	11.14	11	42	11	0.02	11	0	0.01	11	0	0.01
inithx.i.1.col	864	18,707	5.02	54	150	56	33.6	56	0	15.86	56	0	14.75
inithx.i.2.col	645	13,979	6.73	31	338	33	102.06	33	0	53.48	33	0	26.24
inithx.i.3.col	621	13,969	7.26	31	335	33	97.17	33	0	50.37	33	0	25.16
jean.col	80	254	8.04	10	38	12	0.01	12	0	0.00	12	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	600	90	799	†	90	798	†
le450_15a.col	450	8,168	8.09	15	420	15	66.88	15	0	140.04	15	0	22.41
le450_15b.col	450	8,169	8.09	15	427	15	164.63	15	0	368.64	15	0	51.91
le450_15c.col	450	16,680	16.51	15	450	16	600	16	63	†	16	0	268.87
le450_15d.col	450	16,750	16.58	15	450	15	600	15	93	†	15	0	576.79
le450_25a.col	450	8,260	8.18	25	289	25	4.79	25	0	6.30	25	0	1.41
le450_25b.col	450	8,263	8.18	25	314	25	10.36	25	0	14.81	25	0	2.95
le450_25c.col	450	17,343	17.17	25	439	25	441.1	25	0	567.74	25	0	97.94
le450_25d.col	450	17,425	17.25	25	441	25	204.97	25	0	258.32	25	0	52.42
le450_5a.col	450	5,714	5.66	5	450	9	91.96	9	45	†	9	0	168.60
le450_5b.col	450	5,734	5.68	5	450	9	98.95	8	81	†	9	0	178.78
le450_5c.col	450	9,803	9.70	5	450	10	600	10	68	†	10	0	290.80
le450_5d.col	450	9,757	9.66	5	450	10	600	10	71	†	10	0	309.93
miles1000.col	128	3,216	39.57	42	62	45	0.05	45	0	0.02	45	0	0.01
miles1500.col	128	5,198	63.95	73	86	75	20.26	75	0	5.34	75	0	4.90

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
miles250.col	128	387	4.76	8	83	11	0.02	11	0	0.04	11	0	0.02
miles500.col	128	1,170	14.40	20	36	23	0.01	23	0	0.00	23	0	0.00
miles750.col	128	2,113	26.00	31	43	35	0.02	35	0	0.00	35	0	0.01
mug100_1.col	100	166	3.35	3	100	6	1.2	6	0	0.87	6	0	0.24
mug100_25.col	100	166	3.35	3	100	6	1.08	6	0	0.84	6	0	0.25
mug88_1.col	88	146	3.81	3	88	6	0.58	6	0	0.47	6	0	0.14
mug88_25.col	88	146	3.81	3	88	6	0.61	6	0	0.47	6	0	0.15
mulsol.i.1.col	197	3,925	20.33	49	63	51	0.09	51	0	0.04	51	0	0.04
mulsol.i.2.col	188	3,885	22.10	31	122	34	0.3	34	0	0.19	34	0	0.16
mulsol.i.3.col	184	3,916	23.26	31	123	34	0.3	34	0	0.29	34	0	0.17
mulsol.i.4.col	185	3,946	23.18	31	124	34	0.45	34	0	0.42	34	0	0.27
mulsol.i.5.col	186	3,973	23.09	31	125	34	0.36	34	0	0.27	34	0	0.19
myciel3.col	11	20	36.36	2	11	6	0.01	6	0	0.00	6	0	0.00
myciel4.col	23	71	28.06	2	23	6	0.02	6	0	0.00	6	0	0.00
myciel5.col	47	236	21.83	2	47	8	0.03	8	0	0.03	8	0	0.01
myciel6.col	95	755	16.91	2	95	8	1.08	8	0	1.17	8	0	0.31
myciel7.col	191	2,360	13.01	2	191	8	38.1	8	0	34.57	8	0	7.85
qg.order30.col	900	26,100	6.45	30	900	30	600	30	540	†	30	355	†
qg.order40.col	1,600	62,400	4.88	40	1,600	40	600	40	1,235	†	40	1,046	†
qg.order60.col	3,600	212,400	3.28	60	3,600	60	600	60	3,223	†	60	3,034	†
queen10_10.col	100	1,470	29.70	10	100	10	11.06	10	0	11.96	10	0	2.82

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
queen11_11.col	121	1,980	27.27	11	121	11	23.03	11	0	24.27	11	0	5.20
queen12_12.col	144	2,596	25.21	12	144	12	47.1	12	0	53.56	12	0	10.01
queen13_13.col	169	3,328	23.44	13	169	13	89.72	13	0	108.31	13	0	17.84
queen14_14.col	196	4,186	21.91	14	196	14	166.89	14	0	182.74	14	0	29.93
queen15_15.col	225	5,180	20.56	15	225	15	295.73	15	0	330.94	15	0	57.77
queen16_16.col	256	6,320	19.36	16	256	16	513.87	16	0	585.35	16	0	98.20
queen5_5.col	25	160	53.33	5	25	10	0.02	10	0	0.01	10	0	0.00
queen6_6.col	36	290	46.03	6	36	10	0.06	10	0	0.07	10	0	0.03
queen7_7.col	49	476	40.48	7	49	10	0.33	10	0	0.42	10	0	0.12
queen8_12.col	96	1,368	30.00	12	96	12	6.57	12	0	7.47	12	0	1.55
queen8_8.col	64	728	36.11	8	64	10	1.31	10	0	1.48	10	0	0.40
queen9_9.col	81	1,056	32.59	9	81	10	4.04	10	0	4.79	10	0	1.09
r1000.1.col	1,000	14,378	2.88	20	844	23	0.58	20	143	†	23	0	240.71
r1000.1c.col	1,000	485,090	97.12	91	1,000	87	600	87	901	†	87	901	†
r1000.5.col	1,000	238,267	47.70	234	985	74	600	74	723	†	75	722	†
r125.1.col	125	209	2.70	5	122	7	0.02	7	0	1.52	7	0	0.37
r125.1c.col	125	7,501	96.79	46	125	70	600	77	37	†	77	37	†
r125.5.col	125	3,838	49.52	36	120	39	11.05	39	0	7.84	39	0	4.56
r250.1.col	250	867	2.79	8	203	11	0.01	11	0	10.01	11	0	1.58
r250.1c.col	250	30,227	97.12	63	250	78	600	82	157	†	82	157	†
r250.5.col	250	14,849	47.71	65	238	66	600	66	8	†	68	0	511.74

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
school1.col	385	19,095	25.83	14	363	35	600	35	171	†	38	120	†
school1_nsh.col	352	14,612	23.65	14	332	41	600	41	55	†	41	0	550.79
wap01a.col	2,368	110,871	3.96	41	2,281	42	600	42	1,625	†	42	1,462	†
wap02a.col	2,464	111,742	3.68	40	2,372	40	600	40	1,639	†	40	1,509	†
wap03a.col	4,730	286,722	2.56	40	4,702	39	600	37	4,090	†	40	3,858	†
wap04a.col	5,231	294,902	2.16	40	5,207	33	600	33	4,392	†	40	3,987	†
wap05a.col	905	43,081	10.53	50	685	40	600	40	253	†	50	0	489.02
wap06a.col	947	43,571	9.73	40	846	41	600	41	344	†	41	247	†
wap07a.col	1,809	103,368	6.32	40	1,719	40	600	40	1,192	†	41	1,087	†
wap08a.col	1,870	104,176	5.96	40	1,773	40	600	40	1,183	†	41	1,051	†
will199GPIA.col	701	6,772	2.76	6	701	12	7.86	12	330	†	12	168	†
zeroin.i.1.col	211	4,100	18.51	49	79	51	17.47	51	0	10.76	51	0	11.24
zeroin.i.2.col	211	3,541	15.98	30	136	32	4.04	32	0	3.07	32	0	2.51
zeroin.i.3.col	206	3,540	16.77	30	136	32	4.01	32	0	3.05	32	0	2.52

Table A.7: RDS performance for 5-plex.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	25	†	25	142	†	28	136	†
brock200_2.clq	200	9,876	49.63	12	200	16	†	16	124	†	18	108	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_3.clq	200	12,048	60.54	15	200	20	†	20	135	†	21	130	†
brock200_4.clq	200	13,089	65.77	17	200	21	†	21	140	†	23	131	†
brock400_1.clq	400	59,723	74.84	27	400	27	†	27	340	†	27	337	†
brock400_2.clq	400	59,786	74.92	29	400	26	†	26	341	†	26	339	†
brock400_3.clq	400	59,681	74.79	31	400	28	†	28	341	†	28	339	†
brock400_4.clq	400	59,765	74.89	33	400	27	†	27	341	†	27	340	†
brock800_1.clq	800	207,505	64.93	23	800	22	†	22	737	†	23	732	†
brock800_2.clq	800	208,166	65.13	24	800	22	†	22	737	†	22	734	†
brock800_3.clq	800	207,333	64.87	25	800	21	†	21	736	†	22	732	†
brock800_4.clq	800	207,643	64.97	26	800	20	†	20	738	†	21	732	†
c-fat200-1.clq	200	1,534	7.71	12	200	14	0.01	14	0	42.52	14	0	9.44
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.55	24	0	15.58	24	0	5.33
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.81	58	0	1.19	58	0	0.86
c-fat500-1.clq	500	4,459	3.57	14	500	15	0.05	9	280	†	14	71	†
c-fat500-10.clq	500	46,627	37.38	126	500	126	3.17	126	0	5.57	126	0	3.81
c-fat500-2.clq	500	9,139	7.33	26	500	26	1.12	16	198	†	26	0	209.37
c-fat500-5.clq	500	23,191	18.59	64	500	64	1.33	64	0	23.78	64	0	8.36
hamming10-2.clq	1,024	518,656	99.02	512	1,024	55	†	60	943	†	61	942	†
hamming10-4.clq	1,024	434,176	82.89	40	1,024	16	†	16	929	†	17	921	†
hamming6-2.clq	64	1,824	90.48	32	64	48	24.62	48	0	9.96	48	0	9.12
hamming6-4.clq	64	704	34.92	4	64	12	1.65	12	0	1.64	12	0	0.60

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
hamming8-2.clq	256	31,616	96.86	128	256	55	†	60	175	†	61	174	†
hamming8-4.clq	256	20,864	63.92	16	256	16	†	16	163	†	17	153	†
johnson16-2-4.clq	120	5,460	76.47	8	120	20	†	20	58	†	20	57	†
johnson32-2-4.clq	496	107,880	87.88	16	496	26	†	26	349	†	26	349	†
johnson8-2-4.clq	28	210	55.56	4	28	12	0.05	12	0	0.04	12	0	0.03
johnson8-4-4.clq	70	1,855	76.81	14	70	24	†	24	12	†	25	9	†
keller4.clq	171	9,435	64.91	11	171	18	†	18	90	†	19	83	†
keller5.clq	776	225,990	75.16	27	776	18	†	18	695	†	19	689	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	18	†	18	3,280	†	19	3,274	†
MANN_a27.clq	378	70,551	99.02	126	378	351	†	351	21	†	351	21	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	990	†	990	41	†	990	41	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	1,843	†	3,240	78	†	3,240	78	†
MANN_a9.clq	45	918	92.73	16	45	45	0.01	45	0	0.00	45	0	0.00
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	15	†	15	861	†	15	836	†
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	26	†	26	905	†	28	898	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	35	†	35	928	†	36	926	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	14	†	14	1,374	†	15	1,342	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	28	†	29	1,414	†	33	1,402	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	33	†	35	1,433	†	37	1,428	†
p_hat300-1.clq	300	10,933	24.38	8	300	14	†	14	173	†	15	138	†
p_hat300-2.clq	300	21,928	48.89	25	300	28	†	29	205	†	30	198	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat300-3.clq	300	33,390	74.45	36	300	32	†	35	230	†	35	228	†
p_hat500-1.clq	500	31,569	25.31	9	500	14	†	14	366	†	14	341	†
p_hat500-2.clq	500	62,946	50.46	36	500	30	†	30	403	†	31	398	†
p_hat500-3.clq	500	93,800	75.19	50	500	36	†	37	428	†	37	427	†
p_hat700-1.clq	700	60,999	24.93	11	700	13	†	13	569	†	14	527	†
p_hat700-2.clq	700	121,728	49.76	44	700	26	†	26	603	†	27	593	†
p_hat700-3.clq	700	183,010	74.81	62	700	30	†	30	633	†	31	630	†
san1000.clq	1,000	250,500	50.15	15	1,000	39	†	39	906	†	39	905	†
san200_0.7_1.clq	200	13,930	70.00	30	200	73	†	73	51	†	73	50	†
san200_0.7_2.clq	200	13,930	70.00	18	200	56	†	57	96	†	57	96	†
san200_0.9_1.clq	200	17,910	90.00	70	200	40	†	40	143	†	41	141	†
san200_0.9_2.clq	200	17,910	90.00	60	200	46	†	47	136	†	48	133	†
san200_0.9_3.clq	200	17,910	90.00	44	200	43	†	45	138	†	45	138	†
san400_0.5_1.clq	400	39,900	50.00	13	400	31	†	31	318	†	31	318	†
san400_0.7_1.clq	400	55,860	70.00	40	400	90	†	90	214	†	90	214	†
san400_0.7_2.clq	400	55,860	70.00	30	400	56	†	56	281	†	56	281	†
san400_0.7_3.clq	400	55,860	70.00	22	400	54	†	54	272	†	54	271	†
san400_0.9_1.clq	400	71,820	90.00	100	400	41	†	42	341	†	42	339	†
sanr200_0.7.clq	200	13,868	69.69	18	200	24	†	24	138	†	24	135	†
sanr200_0.9.clq	200	17,863	89.76	42	200	43	†	43	140	†	43	140	†
sanr400_0.5.clq	400	39,984	50.11	13	400	3	1.00	17	320	†	18	308	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
sanr400_0.7.clq	400	55,869	70.01	21	400	4	1.00	25	339	†	25	339	†
adjnoun.graph	112	425	6.84	5	112	5	0.90	10	0	12.45	10	0	3.58
as-22july06.graph	22,963	48,436	0.02	17	232	6	0.90	24	0	178.36	24	0	46.29
astro-ph.graph	16,706	121,251	0.09	57	165	7	0.86	57	0	22.65	57	0	7.27
caidaRouterLevel.graph	192,244	609,066	0.00	17	6,447	8	0.86	8	6,216	†	9	6,125	†
celegans_metabolic.graph	453	2,025	1.98	9	313	9	0.83	14	25	†	14	0	209.35
celegansneural.graph	297	2,148	4.89	8	278	10	0.83	12	35	†	14	0	281.51
chesapeake.graph	39	170	22.94	5	39	11	0.82	11	0	0.05	11	0	0.02
cnr-2000.graph	325,557	2,738,969	0.01	84	286	12	0.82	80	186	†	80	177	†
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	13	0.81	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	14	0.81	115	0	0.00	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	164	15	0.80	20	0	11.92	20	0	2.37
cond-mat-2003.graph	31,163	120,029	0.03	25	77	16	0.80	27	0	0.67	27	0	0.26
cond-mat-2005.graph	40,421	175,691	0.02	30	83	17	0.79	30	0	1.42	30	0	0.29
dolphins.graph	62	159	8.41	5	62	18	0.79	9	0	0.52	9	0	0.15
email.graph	1,133	5,451	0.85	12	434	19	0.79	12	204	†	13	109	†
football.graph	115	613	9.35	9	115	20	0.79	12	0	9.77	12	0	2.02
hep-th.graph	8,361	15,751	0.05	24	24	21	0.79	24	0	0.00	24	0	0.00
jazz.graph	198	2,742	14.06	30	30	22	0.79	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	34	23	0.78	9	0	0.02	9	0	0.02
lesmis.graph	77	254	8.68	10	38	24	0.78	12	0	0.04	12	0	0.02

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
memplus.graph	17,758	54,196	0.03	97	97	25	0.78	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	26	0.78	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	172	27	0.78	35	0	2.48	35	0	0.58
polblogs.graph	1,490	16,715	1.51	20	541	28	0.78	20	309	†	23	202	†
polbooks.graph	105	441	8.08	6	105	29	0.78	11	0	16.14	11	0	2.85
power.graph	4,941	6,594	0.05	6	4,941	30	0.78	7	4,754	†	7	4,684	†
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	6,441	31	0.77	15	6,265	†	16	6,164	†
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	1,995	32	0.77	19	1,802	†	19	1,724	†
Cit-HepTh.txt	27,769	352,285	0.09	23	8,595	35	0.77	7	8,408	†	12	8,210	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,498	36	0.77	8	2,277	†	8	2,209	†
Slashdot0811.txt	77,360	469,180	0.02	26	7,397	40	0.77	6	7,207	†	6	7,136	†
Slashdot0902.txt	82,168	504,230	0.02	27	7,368	41	0.77	6	7,180	†	6	7,120	†
soc-Epinions1.txt	75,879	405,740	0.01	23	6,010	42	0.77	17	5,806	†	17	5,733	†
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	43	0.77	201	98	†	201	40	†
web-Google.txt	875,713	4,322,051	0.00	44	223	44	0.77	48	0	139.91	48	0	30.93
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	45	0.77	150	1,118	†	150	1,044	†
web-Stanford.txt	281,903	1,992,636	0.01	61	1,595	46	0.77	24	1,246	†	25	1,212	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,604	47	0.77	10	2,392	†	15	2,211	†
1-FullIns_3.col	30	100	22.99	3	30	48	0.77	8	0	0.04	8	0	0.01
1-FullIns_4.col	93	593	13.86	3	93	49	0.77	10	0	13.58	10	0	2.56
1-FullIns_5.col	282	3,247	8.20	3	282	50	0.77	10	76	†	11	0	590.38

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
1-Insertions_4.col	67	232	10.49	2	67	8	1.86	8	0	2.55	8	0	0.46
1-Insertions_5.col	202	1,227	6.04	2	202	9	2.87	9	0	589.99	9	0	91.10
1-Insertions_6.col	607	6,337	3.45	2	607	6	†	6	418	†	9	273	†
2-FullIns_3.col	52	201	15.16	4	52	9	0.02	9	0	0.66	9	0	0.13
2-FullIns_4.col	212	1,621	7.25	4	212	11	4.10	10	21	†	11	0	149.30
2-FullIns_5.col	852	12,201	3.37	4	852	7	†	7	659	†	10	514	†
2-Insertions_3.col	37	72	10.81	2	37	7	0.06	7	0	0.06	7	0	0.03
2-Insertions_4.col	149	541	4.91	2	149	8	196.58	8	0	159.17	8	0	31.18
2-Insertions_5.col	597	3,936	2.21	2	597	9	227.98	8	375	†	8	315	†
3-FullIns_3.col	80	346	10.95	5	80	10	0.02	10	0	4.46	10	0	1.04
3-FullIns_4.col	405	3,524	4.31	5	405	11	47.63	10	198	†	10	133	†
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	7	†	7	1,839	†	7	1,773	†
3-Insertions_3.col	56	110	7.14	2	56	7	0.72	7	0	0.65	7	0	0.20
3-Insertions_4.col	281	1,046	2.66	2	281	8	†	8	96	†	8	21	†
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	6	†	6	1,223	†	6	1,150	†
4-FullIns_3.col	114	541	8.40	6	114	10	0.05	10	0	48.65	10	0	6.78
4-FullIns_4.col	690	6,650	2.80	6	690	12	390.88	10	468	†	10	402	†
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	7	†	7	3,962	†	7	3,890	†
4-Insertions_3.col	79	156	5.06	2	79	7	5.09	7	0	5.85	7	0	1.18
4-Insertions_4.col	475	1,795	1.59	2	475	8	†	8	285	†	8	213	†
5-FullIns_3.col	154	792	6.72	7	154	11	0.19	11	0	165.48	11	0	24.91

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	10	†	10	846	†	10	777	†
abb313GPIA.col	1,557	53,356	4.41	8	1,555	23	†	23	1,501	†	23	1,500	†
anna.col	138	493	5.22	11	44	13	0.05	13	0	0.08	13	0	0.05
ash331GPIA.col	662	4,181	1.91	3	662	10	15.79	10	485	†	10	406	†
ash608GPIA.col	1,216	7,844	1.06	3	1,216	10	15.38	10	1,038	†	10	962	†
ash958GPIA.col	1,916	12,506	0.68	3	1,916	10	37.35	10	1,738	†	10	1,664	†
C2000.5.col	2,000	999,836	50.02	16	2,000	16	†	16	1,918	†	17	1,906	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	17	†	17	3,919	†	17	3,910	†
david.col	87	406	10.85	11	44	14	0.03	14	0	0.03	14	0	0.02
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	9	†	9	827	†	9	764	†
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	17	†	17	920	†	17	913	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	43	†	45	938	†	46	937	†
DSJC125.1.col	125	736	9.50	4	125	9	2.51	9	0	74.91	9	0	13.48
DSJC125.5.col	125	3,891	50.21	10	125	17	†	17	45	†	17	34	†
DSJC125.9.col	125	6,961	89.82	34	125	43	†	45	63	†	45	63	†
DSJC250.1.col	250	3,218	10.34	4	250	9	†	9	77	†	10	0	570.64
DSJC250.5.col	250	15,668	50.34	12	250	18	†	18	172	†	18	163	†
DSJC250.9.col	250	27,897	89.63	43	250	42	†	42	190	†	43	189	†
DSJC500.1.col	500	12,458	9.99	5	500	9	†	9	334	†	10	247	†
DSJC500.5.col	500	62,624	50.20	13	500	18	†	17	426	†	18	410	†
DSJC500.9.col	500	112,437	90.13	56	500	44	†	46	435	†	46	434	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
DSJR500.1.col	500	3,555	2.85	11	489	16	0.05	10	277	†	14	104	†
DSJR500.1c.col	500	121,275	97.21	83	500	100	†	102	390	†	102	390	†
DSJR500.5.col	500	58,862	47.18	122	492	56	†	59	272	†	60	268	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	18	†	18	911	†	18	900	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	18	†	18	914	†	18	907	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	17	†	17	919	†	17	908	†
flat300_20_0.col	300	21,375	47.66	11	300	17	†	17	216	†	17	205	†
flat300_26_0.col	300	21,633	48.23	11	300	17	†	17	218	†	17	209	†
flat300_28_0.col	300	21,695	48.37	12	300	17	†	17	213	†	17	203	†
fpsol2.i.1.col	496	11,654	9.49	65	120	67	35.26	67	0	13.71	67	0	8.07
fpsol2.i.2.col	451	8,691	8.57	30	260	12	†	18	30	†	33	0	259.80
fpsol2.i.3.col	425	8,688	9.64	30	260	13	†	33	0	536.66	33	0	203.59
games120.col	120	638	8.94	9	120	12	0.02	12	0	13.27	12	0	2.90
homer.col	561	1,628	1.04	13	98	15	0.62	15	0	5.01	15	0	1.07
huck.col	74	301	11.14	11	45	13	0.05	13	0	0.06	13	0	0.03
inithx.i.1.col	864	18,707	5.02	54	158	57	152.34	57	0	67.01	57	0	48.76
inithx.i.2.col	645	13,979	6.73	31	396	12	†	12	165	†	13	98	†
inithx.i.3.col	621	13,969	7.26	31	396	11	†	11	182	†	11	119	†
jean.col	80	254	8.04	10	38	12	0.02	12	0	0.03	12	0	0.02
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	800	†	90	800	†
le450_15a.col	450	8,168	8.09	15	427	13	†	11	237	†	13	124	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
le450_15b.col	450	8,169	8.09	15	429	13	†	12	225	†	14	115	†
le450_15c.col	450	16,680	16.51	15	450	13	†	13	275	†	14	226	†
le450_15d.col	450	16,750	16.58	15	450	12	†	12	287	†	13	225	†
le450_25a.col	450	8,260	8.18	25	297	25	122.77	25	0	196.70	25	0	32.35
le450_25b.col	450	8,263	8.18	25	320	25	329.21	25	0	571.94	25	0	89.52
le450_25c.col	450	17,343	17.17	25	442	16	†	16	244	†	24	73	†
le450_25d.col	450	17,425	17.25	25	442	19	†	16	241	†	21	107	†
le450_5a.col	450	5,714	5.66	5	450	10	†	8	274	†	10	160	†
le450_5b.col	450	5,734	5.68	5	450	10	†	9	260	†	9	193	†
le450_5c.col	450	9,803	9.70	5	450	10	†	9	276	†	10	197	†
le450_5d.col	450	9,757	9.66	5	450	9	†	9	277	†	10	197	†
miles1000.col	128	3,216	39.57	42	81	46	1.11	46	0	0.64	46	0	0.49
miles1500.col	128	5,198	63.95	73	88	76	59.58	76	0	14.50	76	0	13.30
miles250.col	128	387	4.76	8	102	12	0.01	12	0	2.41	12	0	0.65
miles500.col	128	1,170	14.40	20	36	24	0.03	24	0	0.01	24	0	0.01
miles750.col	128	2,113	26.00	31	43	36	0.02	36	0	0.01	36	0	0.01
mug100_1.col	100	166	3.35	3	100	7	18.38	7	0	15.07	7	0	4.02
mug100_25.col	100	166	3.35	3	100	7	18.38	7	0	14.63	7	0	3.99
mug88_1.col	88	146	3.81	3	88	7	8.89	7	0	7.21	7	0	1.98
mug88_25.col	88	146	3.81	3	88	7	9.03	7	0	7.29	7	0	1.92
mulsol.i.1.col	197	3,925	20.33	49	65	52	0.11	52	0	0.04	52	0	0.06

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
mulsol.i.2.col	188	3,885	22.10	31	124	34	1.62	34	0	1.22	34	0	0.99
mulsol.i.3.col	184	3,916	23.26	31	125	34	1.67	34	0	1.71	34	0	1.09
mulsol.i.4.col	185	3,946	23.18	31	126	34	1.76	34	0	1.98	34	0	1.16
mulsol.i.5.col	186	3,973	23.09	31	127	34	1.79	34	0	1.95	34	0	0.89
myciel3.col	11	20	36.36	2	11	8	0.02	8	0	0.00	8	0	0.00
myciel4.col	23	71	28.06	2	23	8	0.02	8	0	0.00	8	0	0.01
myciel5.col	47	236	21.83	2	47	9	0.30	9	0	0.41	9	0	0.13
myciel6.col	95	755	16.91	2	95	10	11.90	10	0	13.36	10	0	3.63
myciel7.col	191	2,360	13.01	2	191	10	†	10	3	†	10	0	169.58
qg.order30.col	900	26,100	6.45	30	900	30	†	30	718	†	30	643	†
qg.order40.col	1,600	62,400	4.88	40	1,600	40	†	40	1,411	†	40	1,344	†
qg.order60.col	3,600	212,400	3.28	60	3,600	60	†	60	3,399	†	60	3,323	†
queen10_10.col	100	1,470	29.70	10	100	13	177.56	13	0	236.22	13	0	35.75
queen11_11.col	121	1,980	27.27	11	121	13	509.99	13	1	†	13	0	94.85
queen12_12.col	144	2,596	25.21	12	144	13	†	13	24	†	13	0	222.38
queen13_13.col	169	3,328	23.44	13	169	13	†	13	47	†	13	0	†
queen14_14.col	196	4,186	21.91	14	196	14	†	14	71	†	14	16	†
queen15_15.col	225	5,180	20.56	15	225	15	†	15	95	†	15	47	†
queen16_16.col	256	6,320	19.36	16	256	16	†	16	123	†	16	75	†
queen5_5.col	25	160	53.33	5	25	13	0.02	13	0	0.01	13	0	0.01
queen6_6.col	36	290	46.03	6	36	13	0.22	13	0	0.45	13	0	0.12

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
queen7_7.col	49	476	40.48	7	49	13	2.01	13	0	3.00	13	0	0.73
queen8_12.col	96	1,368	30.00	12	96	13	121.90	13	0	136.39	13	0	25.28
queen8_8.col	64	728	36.11	8	64	13	11.76	13	0	14.56	13	0	3.51
queen9_9.col	81	1,056	32.59	9	81	13	50.14	13	0	59.11	13	0	12.33
r1000.1.col	1,000	14,378	2.88	20	924	24	4.31	12	669	†	15	538	†
r1000.1c.col	1,000	485,090	97.12	91	1,000	93	†	95	896	†	96	895	†
r1000.5.col	1,000	238,267	47.70	234	985	74	†	74	729	†	75	727	†
r125.1.col	125	209	2.70	5	125	9	0.02	9	0	51.94	9	0	9.03
r125.1c.col	125	7,501	96.79	46	125	89	†	89	27	†	90	26	†
r125.5.col	125	3,838	49.52	36	122	40	127.92	40	0	104.52	40	0	54.08
r250.1.col	250	867	2.79	8	234	11	0.02	11	22	†	11	0	153.44
r250.1c.col	250	30,227	97.12	63	250	95	†	98	144	†	99	142	†
r250.5.col	250	14,849	47.71	65	245	59	†	64	28	†	65	20	†
school1.col	385	19,095	25.83	14	363	39	†	39	187	†	39	185	†
school1_nsh.col	352	14,612	23.65	14	333	35	†	35	145	†	38	103	†
wap01a.col	2,368	110,871	3.96	41	2,288	28	†	28	2,023	†	30	1,949	†
wap02a.col	2,464	111,742	3.68	40	2,377	31	†	31	1,984	†	31	1,931	†
wap03a.col	4,730	286,722	2.56	40	4,717	29	†	29	4,479	†	30	4,415	†
wap04a.col	5,231	294,902	2.16	40	5,223	29	†	17	4,879	†	29	4,610	†
wap05a.col	905	43,081	10.53	50	693	36	†	36	463	†	40	374	†
wap06a.col	947	43,571	9.73	40	865	35	†	35	597	†	40	476	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
wap07a.col	1,809	103,368	6.32	40	1,724	27	†	27	1,527	†	39	1,304	†
wap08a.col	1,870	104,176	5.96	40	1,779	37	†	36	1,409	†	39	1,302	†
will199GPIA.col	701	6,772	2.76	6	701	14	63.29	14	514	†	14	446	†
zeroin.i.1.col	211	4,100	18.51	49	91	52	225.88	52	0	145.00	52	0	141.88
zeroin.i.2.col	211	3,541	15.98	30	137	33	45.38	33	0	33.00	33	0	25.50
zeroin.i.3.col	206	3,540	16.77	30	137	33	45.88	33	0	32.96	33	0	26.17

Table A.8: RDS performance for 1-defective clique.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	20	†	20	43	†	21	23	†
brock200_2.clq	200	9,876	49.63	12	200	12	1.69	12	0	1.79	12	0	0.59
brock200_3.clq	200	12,048	60.54	15	200	15	33.59	15	0	33.96	15	0	10.69
brock200_4.clq	200	13,089	65.77	17	200	17	143.79	17	0	139.93	17	0	47.78
brock400_1.clq	400	59,723	74.84	27	400	22	†	22	232	†	22	221	†
brock400_2.clq	400	59,786	74.92	29	400	20	†	21	239	†	21	218	†
brock400_3.clq	400	59,681	74.79	31	400	20	†	20	242	†	21	219	†
brock400_4.clq	400	59,765	74.89	33	400	19	†	19	248	†	21	230	†
brock800_1.clq	800	207,505	64.93	23	800	16	†	16	568	†	17	521	†
brock800_2.clq	800	208,166	65.13	24	800	18	†	18	565	†	18	531	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock800_3.clq	800	207,333	64.87	25	800	17	†	17	556	†	17	529	†
brock800_4.clq	800	207,643	64.97	26	800	17	†	17	560	†	18	521	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.02	12	0	0.00	12	0	0.01
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.02	24	0	0.00	24	0	0.01
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.01	58	0	0.00	58	0	0.01
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.01	14	0	0.03	14	0	0.03
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.03	126	0	0.01	126	0	0.03
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.02	26	0	0.02	26	0	0.03
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.01	64	0	0.01	64	0	0.03
hamming10-2.clq	1,024	518,656	99.02	512	1,024	512	8.36	512	0	1.11	512	0	1.34
hamming10-4.clq	1,024	434,176	82.89	40	1,024	17	†	17	708	†	17	701	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.01	32	0	0.00	32	0	0.00
hamming6-4.clq	64	704	34.92	4	64	4	0.02	4	0	0.00	4	0	0.00
hamming8-2.clq	256	31,616	96.86	128	256	128	0.06	128	0	0.02	128	0	0.03
hamming8-4.clq	256	20,864	63.92	16	256	16	0.14	16	0	0.19	16	0	0.07
johnson16-2-4.clq	120	5,460	76.47	8	120	8	29.19	8	0	26.59	8	0	11.51
johnson32-2-4.clq	496	107,880	87.88	16	496	8	†	8	314	†	8	311	†
johnson8-2-4.clq	28	210	55.56	4	28	4	0.01	4	0	0.00	4	0	0.00
johnson8-4-4.clq	70	1,855	76.81	14	70	14	0.01	14	0	0.00	14	0	0.00
keller4.clq	171	9,435	64.91	11	171	12	3.68	12	0	3.57	12	0	1.24
keller5.clq	776	225,990	75.16	27	776	16	†	16	488	†	16	465	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
keller6.clq	3,361	4,619,898	81.82	59	3,361	16	†	16	3,074	†	16	3,050	†
MANN_a27.clq	378	70,551	99.02	126	378	21	†	22	316	†	21	318	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	21	†	22	973	†	22	973	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	21	†	22	3,259	†	21	3,261	†
MANN_a9.clq	45	918	92.73	16	45	17	0.14	17	0	0.07	17	0	0.13
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	11	145.67	11	0	159.07	11	0	33.00
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	26	†	26	730	†	28	691	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	25	†	26	863	†	29	845	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	12	†	12	257	†	12	0	456.75
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	25	†	25	1,243	†	26	1,194	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	29	†	29	1,352	†	29	1,347	†
p_hat300-1.clq	300	10,933	24.38	8	300	9	0.30	9	0	0.32	9	0	0.10
p_hat300-2.clq	300	21,928	48.89	25	300	26	482.04	26	0	313.06	26	0	174.29
p_hat300-3.clq	300	33,390	74.45	36	300	26	†	27	155	†	28	144	†
p_hat500-1.clq	500	31,569	25.31	9	500	10	3.90	10	0	4.21	10	0	1.18
p_hat500-2.clq	500	62,946	50.46	36	500	32	†	32	192	†	32	182	†
p_hat500-3.clq	500	93,800	75.19	50	500	30	†	30	363	†	30	358	†
p_hat700-1.clq	700	60,999	24.93	11	700	12	10.72	12	0	13.10	12	0	3.12
p_hat700-2.clq	700	121,728	49.76	44	700	26	†	27	401	†	27	386	†
p_hat700-3.clq	700	183,010	74.81	62	700	29	†	29	536	†	31	521	†
san1000.clq	1,000	250,500	50.15	15	1,000	10	†	10	841	†	10	828	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
san200_0.7_1.clq	200	13,930	70.00	30	200	18	†	20	86	†	20	82	†
san200_0.7_2.clq	200	13,930	70.00	18	200	15	†	15	102	†	15	99	†
san200_0.9_1.clq	200	17,910	90.00	70	200	37	†	37	103	†	37	103	†
san200_0.9_2.clq	200	17,910	90.00	60	200	30	†	31	112	†	31	111	†
san200_0.9_3.clq	200	17,910	90.00	44	200	28	†	28	112	†	29	109	†
san400_0.5_1.clq	400	39,900	50.00	13	400	10	†	10	195	†	10	178	†
san400_0.7_1.clq	400	55,860	70.00	40	400	20	†	20	314	†	20	313	†
san400_0.7_2.clq	400	55,860	70.00	30	400	17	†	17	295	†	17	294	†
san400_0.7_3.clq	400	55,860	70.00	22	400	16	†	16	271	†	16	265	†
san400_0.9_1.clq	400	71,820	90.00	100	400	40	†	40	295	†	40	295	†
sanr200_0.7.clq	200	13,868	69.69	18	200	19	†	19	0	535.85	19	0	208.09
sanr200_0.9.clq	200	17,863	89.76	42	200	27	†	27	120	†	28	117	†
sanr400_0.5.clq	400	39,984	50.11	13	400	14	177.92	14	0	180.10	14	0	45.77
sanr400_0.7.clq	400	55,869	70.01	21	400	21	†	21	184	†	21	164	†
adjnoun.graph	112	425	6.84	5	89	6	0.02	6	0	0.00	6	0	0.00
as-22july06.graph	22,963	48,436	0.02	17	168	18	0.02	18	0	0.01	18	0	0.01
astro-ph.graph	16,706	121,251	0.09	57	113	57	0.01	57	0	0.00	57	0	0.01
caidaRouterLevel.graph	192,244	609,066	0.00	17	4,021	18	1.78	18	0	48.74	18	0	7.65
celegans_metabolic.graph	453	2,025	1.98	9	92	10	0.02	10	0	0.00	10	0	0.00
celegansneural.graph	297	2,148	4.89	8	251	8	0.02	8	0	0.02	8	0	0.01
chesapeake.graph	39	170	22.94	5	39	6	0.02	6	0	0.00	6	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
cnr-2000.graph	325,557	2,738,969	0.01	84	86	85	0.01	85	0	0.00	85	0	0.01
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.01	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.02	115	0	0.01	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	18	18	0.02	18	0	0.00	18	0	0.00
cond-mat-2003.graph	31,163	120,029	0.03	25	27	25	0.01	25	0	0.00	25	0	0.00
cond-mat-2005.graph	40,421	175,691	0.02	30	30	30	0.01	30	0	0.00	30	0	0.00
dolphins.graph	62	159	8.41	5	45	6	0.01	6	0	0.00	6	0	0.00
email.graph	1,133	5,451	0.85	12	121	12	0.02	12	0	0.01	12	0	0.01
football.graph	115	613	9.35	9	115	9	0.02	9	0	0.00	9	0	0.00
hep-th.graph	8,361	15,751	0.05	24	24	24	0.01	24	0	0.00	24	0	0.00
jazz.graph	198	2,742	14.06	30	30	30	0.02	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	22	6	0.02	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	20	10	0.01	10	0	0.00	10	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.01	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.02	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	126	26	0.03	26	0	0.02	26	0	0.02
polblogs.graph	1,490	16,715	1.51	20	459	21	0.34	21	0	0.39	21	0	0.18
polbooks.graph	105	441	8.08	6	98	7	0.01	7	0	0.00	7	0	0.01
power.graph	4,941	6,594	0.05	6	36	6	0.02	6	0	0.00	6	0	0.00
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	125	15	0.01	15	0	0.00	15	0	0.00
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	55	19	0.01	19	0	0.00	19	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
Cit-HepTh.txt	27,769	352,285	0.09	23	7,280	24	128.52	23	558	†	24	0	90.70
Email-EuAll.txt	265,009	364,481	0.00	16	1,883	17	1.56	17	0	4.94	17	0	1.06
Slashdot0811.txt	77,360	469,180	0.02	26	6,139	27	64.10	27	0	184.56	27	0	24.95
Slashdot0902.txt	82,168	504,230	0.02	27	6,086	28	31.87	28	0	137.74	28	0	20.78
soc-Epinions1.txt	75,879	405,740	0.01	23	5,243	24	289.06	24	0	383.15	24	0	84.87
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	0.14	202	0	0.06	202	0	0.04
web-Google.txt	875,713	4,322,051	0.00	44	218	44	†	45	0	0.02	45	0	0.02
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	155	4.73	155	0	6.00	155	0	1.15
web-Stanford.txt	281,903	1,992,636	0.01	61	1,389	59	†	51	310	†	51	310	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,382	18	9.13	18	0	26.58	18	0	6.22
1-FullIns_3.col	30	100	22.99	3	30	4	0.01	4	0	0.00	4	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	4	0.01	4	0	0.00	4	0	0.00
1-FullIns_5.col	282	3,247	8.20	3	282	4	0.03	4	0	0.04	4	0	0.02
1-Insertions_4.col	67	232	10.49	2	67	3	0.01	3	0	0.00	3	0	0.00
1-Insertions_5.col	202	1,227	6.04	2	202	3	0.02	3	0	0.02	3	0	0.01
1-Insertions_6.col	607	6,337	3.45	2	607	3	0.11	3	0	0.30	3	0	0.09
2-FullIns_3.col	52	201	15.16	4	52	5	0.02	5	0	0.00	5	0	0.00
2-FullIns_4.col	212	1,621	7.25	4	212	5	0.01	5	0	0.02	5	0	0.01
2-FullIns_5.col	852	12,201	3.37	4	852	5	0.19	5	0	0.62	5	0	0.19
2-Insertions_3.col	37	72	10.81	2	37	3	0.02	3	0	0.00	3	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	3	0.01	3	0	0.01	3	0	0.01

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
2-Insertions_5.col	597	3,936	2.21	2	597	3	0.02	3	0	0.25	3	0	0.08
3-FullIns_3.col	80	346	10.95	5	80	6	0.01	6	0	0.00	6	0	0.00
3-FullIns_4.col	405	3,524	4.31	5	405	6	0.02	6	0	0.06	6	0	0.03
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	6	1.31	6	0	16.47	6	0	2.68
3-Insertions_3.col	56	110	7.14	2	56	3	0.02	3	0	0.00	3	0	0.00
3-Insertions_4.col	281	1,046	2.66	2	281	3	0.02	3	0	0.03	3	0	0.02
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	3	0.11	3	0	8.15	3	0	1.41
4-FullIns_3.col	114	541	8.40	6	114	7	0.01	7	0	0.00	7	0	0.00
4-FullIns_4.col	690	6,650	2.80	6	690	7	0.06	7	0	0.50	7	0	0.10
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	7	7.08	7	0	219.54	7	0	26.04
4-Insertions_3.col	79	156	5.06	2	79	3	0.01	3	0	0.00	3	0	0.00
4-Insertions_4.col	475	1,795	1.59	2	475	3	0.02	3	0	0.22	3	0	0.05
5-FullIns_3.col	154	792	6.72	7	136	8	0.01	8	0	0.00	8	0	0.01
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	8	0.16	8	0	2.84	8	0	0.48
abb313GPIA.col	1,557	53,356	4.41	8	1,552	9	24.03	9	0	39.43	9	0	10.27
anna.col	138	493	5.22	11	19	11	0.02	11	0	0.00	11	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	4	0.02	4	0	0.37	4	0	0.10
ash608GPIA.col	1,216	7,844	1.06	3	1,216	3	0.03	3	0	8.99	3	0	0.91
ash958GPIA.col	1,916	12,506	0.68	3	1,916	3	0.06	3	0	30.22	3	0	3.58
C2000.5.col	2,000	999,836	50.02	16	2,000	13	†	13	1,572	†	14	1,439	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	14	†	14	3,528	†	14	3,440	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
david.col	87	406	10.85	11	22	11	0.01	11	0	0.00	11	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	6	3.24	6	0	3.65	6	0	0.70
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	14	†	14	530	†	14	447	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	30	†	31	914	†	31	911	†
DSJC125.1.col	125	736	9.50	4	125	5	0.02	5	0	0.00	5	0	0.01
DSJC125.5.col	125	3,891	50.21	10	125	11	0.25	11	0	0.25	11	0	0.09
DSJC125.9.col	125	6,961	89.82	34	125	28	†	28	43	†	29	40	†
DSJC250.1.col	250	3,218	10.34	4	250	5	0.03	5	0	0.02	5	0	0.02
DSJC250.5.col	250	15,668	50.34	12	250	12	13.35	12	0	14.78	12	0	3.68
DSJC250.9.col	250	27,897	89.63	43	250	29	†	29	160	†	30	157	†
DSJC500.1.col	500	12,458	9.99	5	500	6	0.38	6	0	0.30	6	0	0.09
DSJC500.5.col	500	62,624	50.20	13	500	14	†	14	45	†	14	0	278.33
DSJC500.9.col	500	112,437	90.13	56	500	27	†	28	414	†	29	412	†
DSJR500.1.col	500	3,555	2.85	11	328	13	0.02	13	0	0.02	13	0	0.02
DSJR500.1c.col	500	121,275	97.21	83	500	35	†	35	438	†	35	438	†
DSJR500.5.col	500	58,862	47.18	122	488	87	†	123	0	499.46	123	0	293.48
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	13	†	13	547	†	14	408	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	14	†	14	513	†	14	421	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	13	†	13	543	†	15	371	†
flat300_20_0.col	300	21,375	47.66	11	300	12	17.21	12	0	18.10	12	0	4.84
flat300_26_0.col	300	21,633	48.23	11	300	12	22.84	12	0	24.46	12	0	5.97

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
flat300_28_0.col	300	21,695	48.37	12	300	12	23.48	12	0	24.56	12	0	6.34
fpsol2.i.1.col	496	11,654	9.49	65	85	66	0.02	66	0	0.00	66	0	0.01
fpsol2.i.2.col	451	8,691	8.57	30	165	31	0.02	31	0	0.01	31	0	0.01
fpsol2.i.3.col	425	8,688	9.64	30	164	31	0.01	31	0	0.01	31	0	0.01
games120.col	120	638	8.94	9	120	9	0.01	9	0	0.00	9	0	0.00
homer.col	561	1,628	1.04	13	35	13	0.01	13	0	0.00	13	0	0.00
huck.col	74	301	11.14	11	20	11	0.01	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	122	55	1.39	55	0	0.06	55	0	0.08
inithx.i.2.col	645	13,979	6.73	31	226	31	0.09	31	0	0.03	31	0	0.04
inithx.i.3.col	621	13,969	7.26	31	212	31	0.08	31	0	0.03	31	0	0.03
jean.col	80	254	8.04	10	20	10	0.02	10	0	0.00	10	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	774	†	90	771	†
le450_15a.col	450	8,168	8.09	15	414	15	0.03	15	0	0.04	15	0	0.03
le450_15b.col	450	8,169	8.09	15	417	15	0.03	15	0	0.06	15	0	0.03
le450_15c.col	450	16,680	16.51	15	450	15	0.14	15	0	0.23	15	0	0.06
le450_15d.col	450	16,750	16.58	15	450	15	0.16	15	0	0.26	15	0	0.07
le450_25a.col	450	8,260	8.18	25	272	25	0.02	25	0	0.01	25	0	0.01
le450_25b.col	450	8,263	8.18	25	304	25	0.02	25	0	0.02	25	0	0.02
le450_25c.col	450	17,343	17.17	25	436	25	0.08	25	0	0.10	25	0	0.05
le450_25d.col	450	17,425	17.25	25	438	25	0.06	25	0	0.08	25	0	0.04
le450_5a.col	450	5,714	5.66	5	450	6	0.06	6	0	0.12	6	0	0.04

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
le450_5b.col	450	5,734	5.68	5	450	6	0.06	6	0	0.12	6	0	0.03
le450_5c.col	450	9,803	9.70	5	450	6	0.19	6	0	0.14	6	0	0.06
le450_5d.col	450	9,757	9.66	5	450	6	0.16	6	0	0.14	6	0	0.05
miles1000.col	128	3,216	39.57	42	51	43	0.01	43	0	0.00	43	0	0.00
miles1500.col	128	5,198	63.95	73	84	73	0.13	73	0	0.06	73	0	0.04
miles250.col	128	387	4.76	8	27	8	0.02	8	0	0.00	8	0	0.00
miles500.col	128	1,170	14.40	20	29	21	0.01	21	0	0.00	21	0	0.00
miles750.col	128	2,113	26.00	31	39	32	0.01	32	0	0.00	32	0	0.00
mug100_1.col	100	166	3.35	3	100	4	0.02	4	0	0.00	4	0	0.00
mug100_25.col	100	166	3.35	3	100	4	0.02	4	0	0.00	4	0	0.00
mug88_1.col	88	146	3.81	3	88	4	0.01	4	0	0.00	4	0	0.00
mug88_25.col	88	146	3.81	3	88	4	0.01	4	0	0.00	4	0	0.00
mulsol.i.1.col	197	3,925	20.33	49	56	50	0.02	50	0	0.00	50	0	0.00
mulsol.i.2.col	188	3,885	22.10	31	116	31	0.08	31	0	0.04	31	0	0.03
mulsol.i.3.col	184	3,916	23.26	31	117	31	0.08	31	0	0.06	31	0	0.03
mulsol.i.4.col	185	3,946	23.18	31	118	31	0.08	31	0	0.06	31	0	0.03
mulsol.i.5.col	186	3,973	23.09	31	119	31	0.08	31	0	0.05	31	0	0.03
myciel3.col	11	20	36.36	2	11	3	0.02	3	0	0.00	3	0	0.00
myciel4.col	23	71	28.06	2	23	3	0.01	3	0	0.00	3	0	0.00
myciel5.col	47	236	21.83	2	47	3	0.01	3	0	0.00	3	0	0.00
myciel6.col	95	755	16.91	2	95	3	0.02	3	0	0.01	3	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
myciel7.col	191	2,360	13.01	2	191	3	0.05	3	0	0.04	3	0	0.01
qg.order30.col	900	26,100	6.45	30	900	30	1.44	30	0	1.44	30	0	0.33
qg.order40.col	1,600	62,400	4.88	40	1,600	40	7.78	40	0	14.35	40	0	2.08
qg.order60.col	3,600	212,400	3.28	60	3,600	60	87.49	60	0	235.29	60	0	25.64
queen10_10.col	100	1,470	29.70	10	100	10	0.02	10	0	0.01	10	0	0.01
queen11_11.col	121	1,980	27.27	11	121	11	0.02	11	0	0.02	11	0	0.01
queen12_12.col	144	2,596	25.21	12	144	12	0.02	12	0	0.02	12	0	0.01
queen13_13.col	169	3,328	23.44	13	169	13	0.03	13	0	0.04	13	0	0.01
queen14_14.col	196	4,186	21.91	14	196	14	0.05	14	0	0.06	14	0	0.02
queen15_15.col	225	5,180	20.56	15	225	15	0.06	15	0	0.08	15	0	0.02
queen16_16.col	256	6,320	19.36	16	256	16	0.08	16	0	0.12	16	0	0.03
queen5_5.col	25	160	53.33	5	25	5	0.02	5	0	0.00	5	0	0.00
queen6_6.col	36	290	46.03	6	36	6	0.01	6	0	0.00	6	0	0.00
queen7_7.col	49	476	40.48	7	49	7	0.01	7	0	0.00	7	0	0.00
queen8_12.col	96	1,368	30.00	12	96	12	0.02	12	0	0.01	12	0	0.01
queen8_8.col	64	728	36.11	8	64	8	0.02	8	0	0.00	8	0	0.00
queen9_9.col	81	1,056	32.59	9	81	9	0.02	9	0	0.01	9	0	0.00
r1000.1.col	1,000	14,378	2.88	20	463	21	0.02	21	0	0.04	21	0	0.03
r1000.1c.col	1,000	485,090	97.12	91	1,000	32	†	35	936	†	35	936	†
r1000.5.col	1,000	238,267	47.70	234	984	130	†	152	388	†	152	386	†
r125.1.col	125	209	2.70	5	57	6	0.01	6	0	0.00	6	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
r125.1c.col	125	7,501	96.79	46	125	35	†	36	63	†	36	63	†
r125.5.col	125	3,838	49.52	36	119	36	0.13	36	0	0.08	36	0	0.08
r250.1.col	250	867	2.79	8	70	8	0.01	8	0	0.00	8	0	0.00
r250.1c.col	250	30,227	97.12	63	250	37	†	37	185	†	37	185	†
r250.5.col	250	14,849	47.71	65	237	65	10.55	65	0	2.90	65	0	1.82
school1.col	385	19,095	25.83	14	361	15	†	15	68	†	15	64	†
school1_nsh.col	352	14,612	23.65	14	331	15	†	15	0	555.48	15	0	372.61
wap01a.col	2,368	110,871	3.96	41	2,107	41	9.66	41	0	21.39	41	0	5.10
wap02a.col	2,464	111,742	3.68	40	2,248	40	13.39	40	0	39.76	40	0	7.75
wap03a.col	4,730	286,722	2.56	40	4,701	41	†	41	96	†	41	93	†
wap04a.col	5,231	294,902	2.16	40	5,204	41	126.02	41	0	577.81	41	0	90.34
wap05a.col	905	43,081	10.53	50	675	50	0.36	50	0	0.33	50	0	0.19
wap06a.col	947	43,571	9.73	40	807	40	2.82	40	0	2.30	40	0	1.03
wap07a.col	1,809	103,368	6.32	40	1,701	41	12.95	41	0	19.82	41	0	4.68
wap08a.col	1,870	104,176	5.96	40	1,753	40	13.10	40	0	20.78	40	0	4.46
will199GPIA.col	701	6,772	2.76	6	700	7	0.03	7	0	0.50	7	0	0.14
zeroin.i.1.col	211	4,100	18.51	49	73	49	0.09	49	0	0.03	49	0	0.04
zeroin.i.2.col	211	3,541	15.98	30	106	30	0.02	30	0	0.01	30	0	0.01
zeroin.i.3.col	206	3,540	16.77	30	106	30	0.02	30	0	0.01	30	0	0.01

Table A.9: RDS performance for 2-defective clique.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	20	†	20	74	†	21	62	†
brock200_2.clq	200	9,876	49.63	12	200	12	28.77	12	0	32.84	12	0	7.38
brock200_3.clq	200	12,048	60.54	15	200	16	537.02	16	0	537.43	16	0	137.71
brock200_4.clq	200	13,089	65.77	17	200	17	†	17	22	†	18	0	556.82
brock400_1.clq	400	59,723	74.84	27	400	20	†	20	272	†	21	259	†
brock400_2.clq	400	59,786	74.92	29	400	20	†	21	263	†	21	255	†
brock400_3.clq	400	59,681	74.79	31	400	20	†	20	275	†	20	263	†
brock400_4.clq	400	59,765	74.89	33	400	20	†	20	268	†	20	261	†
brock800_1.clq	800	207,505	64.93	23	800	16	†	16	621	†	17	589	†
brock800_2.clq	800	208,166	65.13	24	800	17	†	17	621	†	17	599	†
brock800_3.clq	800	207,333	64.87	25	800	17	†	17	613	†	17	590	†
brock800_4.clq	800	207,643	64.97	26	800	16	†	16	627	†	17	594	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.01	12	0	0.01	12	0	0.01
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.01	24	0	0.00	24	0	0.01
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.01	58	0	0.00	58	0	0.01
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.02	14	0	0.07	14	0	0.08
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.03	126	0	0.02	126	0	0.05
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.01	26	0	0.04	26	0	0.05
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.01	64	0	0.02	64	0	0.04
hamming10-2.clq	1,024	518,656	99.02	512	1,024	512	17.77	512	0	3.11	512	0	3.18

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
hamming10-4.clq	1,024	434,176	82.89	40	1,024	18	†	18	723	†	18	715	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.01	32	0	0.00	32	0	0.00
hamming6-4.clq	64	704	34.92	4	64	5	0.01	5	0	0.01	5	0	0.01
hamming8-2.clq	256	31,616	96.86	128	256	128	0.14	128	0	0.04	128	0	0.05
hamming8-4.clq	256	20,864	63.92	16	256	16	2.62	16	0	2.78	16	0	0.98
johnson16-2-4.clq	120	5,460	76.47	8	120	9	373.06	9	0	327.46	9	0	105.73
johnson32-2-4.clq	496	107,880	87.88	16	496	7	†	8	346	†	8	344	†
johnson8-2-4.clq	28	210	55.56	4	28	5	0.01	5	0	0.00	5	0	0.00
johnson8-4-4.clq	70	1,855	76.81	14	70	14	0.02	14	0	0.01	14	0	0.02
keller4.clq	171	9,435	64.91	11	171	13	61.45	13	0	58.65	13	0	18.50
keller5.clq	776	225,990	75.16	27	776	15	†	15	564	†	15	549	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	15	†	15	3,149	†	15	3,134	†
MANN_a27.clq	378	70,551	99.02	126	378	19	†	20	325	†	20	325	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	19	†	20	982	†	20	982	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	19	†	20	3,268	†	20	3,268	†
MANN_a9.clq	45	918	92.73	16	45	18	1.98	18	0	0.89	18	0	1.34
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	11	†	11	271	†	12	0	482.12
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	24	†	24	785	†	25	776	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	25	†	25	891	†	26	878	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	11	†	11	787	†	12	519	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	23	†	24	1,296	†	24	1,286	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	26	†	27	1,395	†	28	1,389	†
p_hat300-1.clq	300	10,933	24.38	8	300	9	4.52	9	0	4.64	9	0	1.19
p_hat300-2.clq	300	21,928	48.89	25	300	22	†	22	75	†	24	48	†
p_hat300-3.clq	300	33,390	74.45	36	300	26	†	26	176	†	27	165	†
p_hat500-1.clq	500	31,569	25.31	9	500	11	52.95	11	0	54.29	11	0	12.76
p_hat500-2.clq	500	62,946	50.46	36	500	29	†	29	247	†	31	228	†
p_hat500-3.clq	500	93,800	75.19	50	500	29	†	30	380	†	30	373	†
p_hat700-1.clq	700	60,999	24.93	11	700	12	213.99	12	0	276.30	12	0	52.55
p_hat700-2.clq	700	121,728	49.76	44	700	25	†	25	477	†	26	456	†
p_hat700-3.clq	700	183,010	74.81	62	700	28	†	28	561	†	29	558	†
san1000.clq	1,000	250,500	50.15	15	1,000	11	†	11	876	†	11	871	†
san200_0.7_1.clq	200	13,930	70.00	30	200	18	†	18	112	†	18	112	†
san200_0.7_2.clq	200	13,930	70.00	18	200	15	†	15	118	†	15	117	†
san200_0.9_1.clq	200	17,910	90.00	70	200	34	†	37	108	†	38	106	†
san200_0.9_2.clq	200	17,910	90.00	60	200	28	†	28	129	†	29	126	†
san200_0.9_3.clq	200	17,910	90.00	44	200	27	†	27	124	†	27	122	†
san400_0.5_1.clq	400	39,900	50.00	13	400	10	†	10	263	†	10	248	†
san400_0.7_1.clq	400	55,860	70.00	40	400	20	†	21	320	†	21	320	†
san400_0.7_2.clq	400	55,860	70.00	30	400	18	†	18	306	†	18	302	†
san400_0.7_3.clq	400	55,860	70.00	22	400	16	†	16	303	†	16	297	†
san400_0.9_1.clq	400	71,820	90.00	100	400	31	†	33	316	†	35	311	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
sanr200_0.7.clq	200	13,868	69.69	18	200	18	†	18	49	†	19	28	†
sanr200_0.9.clq	200	17,863	89.76	42	200	28	†	28	125	†	28	125	†
sanr400_0.5.clq	400	39,984	50.11	13	400	14	†	14	87	†	14	15	†
sanr400_0.7.clq	400	55,869	70.01	21	400	18	†	18	251	†	19	231	†
adjnoun.graph	112	425	6.84	5	102	6	0.02	6	0	0.01	6	0	0.01
as-22july06.graph	22,963	48,436	0.02	17	182	18	0.03	18	0	0.02	18	0	0.02
astro-ph.graph	16,706	121,251	0.09	57	113	57	0.02	57	0	0.00	57	0	0.00
caidaRouterLevel.graph	192,244	609,066	0.00	17	4,704	19	6.38	19	0	586.23	19	0	202.70
celegans_metabolic.graph	453	2,025	1.98	9	138	10	0.02	10	0	0.01	10	0	0.01
celegansneural.graph	297	2,148	4.89	8	265	9	0.06	9	0	0.12	9	0	0.06
chesapeake.graph	39	170	22.94	5	39	6	0.01	6	0	0.00	6	0	0.00
cnr-2000.graph	325,557	2,738,969	0.01	84	89	85	0.01	85	0	0.00	85	0	0.00
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.01	87	0	0.00	87	0	0.00
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.02	115	0	0.00	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	53	18	0.01	18	0	0.00	18	0	0.00
cond-mat-2003.graph	31,163	120,029	0.03	25	50	25	0.01	25	0	0.00	25	0	0.00
cond-mat-2005.graph	40,421	175,691	0.02	30	30	30	0.01	30	0	0.00	30	0	0.00
dolphins.graph	62	159	8.41	5	53	6	0.02	6	0	0.00	6	0	0.00
email.graph	1,133	5,451	0.85	12	238	12	0.05	12	0	0.14	12	0	0.04
football.graph	115	613	9.35	9	115	9	0.01	9	0	0.00	9	0	0.01
hep-th.graph	8,361	15,751	0.05	24	24	24	0.01	24	0	0.00	24	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
jazz.graph	198	2,742	14.06	30	30	30	0.01	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	33	6	0.01	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	31	11	0.02	11	0	0.00	11	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.02	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	145	27	0.09	27	0	0.07	27	0	0.04
polblogs.graph	1,490	16,715	1.51	20	489	22	2.75	22	0	3.13	22	0	1.23
polbooks.graph	105	441	8.08	6	103	7	0.02	7	0	0.01	7	0	0.01
power.graph	4,941	6,594	0.05	6	231	6	0.01	6	0	0.04	6	0	0.03
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	650	16	0.01	16	0	2.43	16	0	0.58
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	211	19	0.01	19	0	0.11	19	0	0.03
Cit-HepTh.txt	27,769	352,285	0.09	23	7,744	25	†	24	2,914	†	24	1,615	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,051	17	13.96	17	0	40.23	17	0	13.95
Slashdot0811.txt	77,360	469,180	0.02	26	6,571	28	525.32	14	1,767	†	28	0	491.81
Slashdot0902.txt	82,168	504,230	0.02	27	6,532	29	219.57	15	1,561	†	29	0	411.07
soc-Epinions1.txt	75,879	405,740	0.01	23	5,456	23	†	22	2,134	†	23	983	†
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	1.90	202	0	2.06	202	0	0.37
web-Google.txt	875,713	4,322,051	0.00	44	222	44	†	46	0	0.08	46	0	0.09
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	155	319.48	155	0	424.72	155	0	70.17
web-Stanford.txt	281,903	1,992,636	0.01	61	1,439	59	†	52	325	†	52	325	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,452	19	124.88	19	5	†	19	0	133.44

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
1-FullIns_3.col	30	100	22.99	3	30	5	0.01	5	0	0.00	5	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	5	0.01	5	0	0.01	5	0	0.01
1-FullIns_5.col	282	3,247	8.20	3	282	5	0.06	5	0	0.18	5	0	0.10
1-Insertions_4.col	67	232	10.49	2	67	4	0.01	4	0	0.00	4	0	0.00
1-Insertions_5.col	202	1,227	6.04	2	202	4	0.02	4	0	0.07	4	0	0.04
1-Insertions_6.col	607	6,337	3.45	2	607	4	0.11	4	0	2.49	4	0	0.88
2-FullIns_3.col	52	201	15.16	4	52	5	0.01	5	0	0.00	5	0	0.00
2-FullIns_4.col	212	1,621	7.25	4	212	6	0.02	6	0	0.08	6	0	0.04
2-FullIns_5.col	852	12,201	3.37	4	852	6	0.52	6	0	8.42	6	0	2.83
2-Insertions_3.col	37	72	10.81	2	37	4	0.01	4	0	0.00	4	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	4	0.01	4	0	0.02	4	0	0.01
2-Insertions_5.col	597	3,936	2.21	2	597	4	0.03	4	0	1.60	4	0	0.60
3-FullIns_3.col	80	346	10.95	5	80	6	0.01	6	0	0.00	6	0	0.00
3-FullIns_4.col	405	3,524	4.31	5	405	7	0.03	7	0	0.37	7	0	0.16
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	7	1.79	7	0	188.14	7	0	51.90
3-Insertions_3.col	56	110	7.14	2	56	4	0.01	4	0	0.00	4	0	0.00
3-Insertions_4.col	281	1,046	2.66	2	281	4	0.01	4	0	0.11	4	0	0.06
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	4	0.11	4	0	57.93	4	0	15.55
4-FullIns_3.col	114	541	8.40	6	114	7	0.02	7	0	0.01	7	0	0.01
4-FullIns_4.col	690	6,650	2.80	6	690	8	0.09	8	0	2.73	8	0	0.85
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	8	8.49	6	1,562	†	8	0	546.25

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
4-Insertions_3.col	79	156	5.06	2	79	4	0.01	4	0	0.00	4	0	0.00
4-Insertions_4.col	475	1,795	1.59	2	475	4	0.02	4	0	0.64	4	0	0.24
5-FullIns_3.col	154	792	6.72	7	154	8	0.02	8	0	0.03	8	0	0.02
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	9	0.20	9	0	17.65	9	0	5.97
abb313GPIA.col	1,557	53,356	4.41	8	1,552	10	275.61	10	10	†	10	0	147.43
anna.col	138	493	5.22	11	19	11	0.02	11	0	0.00	11	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	4	0.05	4	0	2.75	4	0	1.00
ash608GPIA.col	1,216	7,844	1.06	3	1,216	4	0.09	4	0	32.73	4	0	10.83
ash958GPIA.col	1,916	12,506	0.68	3	1,916	4	0.19	4	0	166.17	4	0	46.59
C2000.5.col	2,000	999,836	50.02	16	2,000	13	†	13	1,697	†	14	1,608	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	13	†	13	3,693	†	14	3,609	†
david.col	87	406	10.85	11	33	11	0.01	11	0	0.00	11	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	7	41.53	7	0	48.11	7	0	11.64
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	13	†	13	696	†	14	604	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	30	†	30	925	†	30	925	†
DSJC125.1.col	125	736	9.50	4	125	5	0.01	5	0	0.01	5	0	0.01
DSJC125.5.col	125	3,891	50.21	10	125	11	1.90	11	0	1.88	11	0	0.62
DSJC125.9.col	125	6,961	89.82	34	125	27	†	27	55	†	27	55	†
DSJC250.1.col	250	3,218	10.34	4	250	6	0.16	6	0	0.14	6	0	0.06
DSJC250.5.col	250	15,668	50.34	12	250	13	118.11	13	0	142.96	13	0	29.95
DSJC250.9.col	250	27,897	89.63	43	250	28	†	30	168	†	30	164	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
DSJC500.1.col	500	12,458	9.99	5	500	6	4.07	6	0	3.43	6	0	0.86
DSJC500.5.col	500	62,624	50.20	13	500	13	†	13	203	†	14	120	†
DSJC500.9.col	500	112,437	90.13	56	500	26	†	28	423	†	28	422	†
DSJR500.1.col	500	3,555	2.85	11	423	14	0.02	14	0	0.09	14	0	0.06
DSJR500.1c.col	500	121,275	97.21	83	500	34	†	36	442	†	36	442	†
DSJR500.5.col	500	58,862	47.18	122	489	69	†	87	150	†	103	78	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	13	†	13	678	†	13	615	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	13	†	13	691	†	14	601	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	13	†	13	681	†	13	616	†
flat300_20_0.col	300	21,375	47.66	11	300	12	369.35	12	0	394.00	12	0	79.87
flat300_26_0.col	300	21,633	48.23	11	300	13	248.60	13	0	266.77	13	0	59.08
flat300_28_0.col	300	21,695	48.37	12	300	13	205.55	13	0	219.48	13	0	49.94
fpsol2.i.1.col	496	11,654	9.49	65	86	66	0.08	66	0	0.02	66	0	0.02
fpsol2.i.2.col	451	8,691	8.57	30	203	31	0.09	31	0	0.05	31	0	0.04
fpsol2.i.3.col	425	8,688	9.64	30	203	31	0.09	31	0	0.04	31	0	0.04
games120.col	120	638	8.94	9	120	9	0.01	9	0	0.00	9	0	0.00
homer.col	561	1,628	1.04	13	61	13	0.01	13	0	0.00	13	0	0.00
huck.col	74	301	11.14	11	32	11	0.01	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	143	55	18.47	55	0	0.57	55	0	0.51
inithx.i.2.col	645	13,979	6.73	31	278	32	0.33	32	0	0.08	32	0	0.06
inithx.i.3.col	621	13,969	7.26	31	268	32	0.31	32	0	0.07	32	0	0.06

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
jean.col	80	254	8.04	10	31	11	0.02	11	0	0.00	11	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	785	†	90	783	†
le450_15a.col	450	8,168	8.09	15	419	15	0.17	15	0	0.23	15	0	0.09
le450_15b.col	450	8,169	8.09	15	421	15	0.28	15	0	0.40	15	0	0.13
le450_15c.col	450	16,680	16.51	15	450	15	1.45	15	0	1.47	15	0	0.37
le450_15d.col	450	16,750	16.58	15	450	15	1.67	15	0	1.99	15	0	0.47
le450_25a.col	450	8,260	8.18	25	280	25	0.03	25	0	0.04	25	0	0.02
le450_25b.col	450	8,263	8.18	25	308	25	0.06	25	0	0.08	25	0	0.04
le450_25c.col	450	17,343	17.17	25	438	25	0.83	25	0	0.98	25	0	0.30
le450_25d.col	450	17,425	17.25	25	440	25	0.47	25	0	0.54	25	0	0.18
le450_5a.col	450	5,714	5.66	5	450	6	0.48	6	0	1.19	6	0	0.33
le450_5b.col	450	5,734	5.68	5	450	6	0.52	6	0	1.26	6	0	0.33
le450_5c.col	450	9,803	9.70	5	450	7	2.34	7	0	2.37	7	0	0.55
le450_5d.col	450	9,757	9.66	5	450	7	2.00	7	0	2.05	7	0	0.49
miles1000.col	128	3,216	39.57	42	61	43	0.02	43	0	0.01	43	0	0.01
miles1500.col	128	5,198	63.95	73	85	73	0.41	73	0	0.10	73	0	0.11
miles250.col	128	387	4.76	8	41	9	0.02	9	0	0.00	9	0	0.00
miles500.col	128	1,170	14.40	20	35	21	0.01	21	0	0.00	21	0	0.00
miles750.col	128	2,113	26.00	31	41	33	0.02	33	0	0.00	33	0	0.00
mug100_1.col	100	166	3.35	3	100	4	0.01	4	0	0.01	4	0	0.00
mug100_25.col	100	166	3.35	3	100	4	0.01	4	0	0.00	4	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
mug88_1.col	88	146	3.81	3	88	4	0.02	4	0	0.00	4	0	0.00
mug88_25.col	88	146	3.81	3	88	4	0.02	4	0	0.00	4	0	0.00
mulsol.i.1.col	197	3,925	20.33	49	57	50	0.02	50	0	0.00	50	0	0.01
mulsol.i.2.col	188	3,885	22.10	31	119	32	0.08	32	0	0.02	32	0	0.03
mulsol.i.3.col	184	3,916	23.26	31	120	32	0.11	32	0	0.05	32	0	0.04
mulsol.i.4.col	185	3,946	23.18	31	121	32	0.09	32	0	0.03	32	0	0.02
mulsol.i.5.col	186	3,973	23.09	31	122	32	0.09	32	0	0.03	32	0	0.03
myciel3.col	11	20	36.36	2	11	4	0.02	4	0	0.00	4	0	0.00
myciel4.col	23	71	28.06	2	23	4	0.01	4	0	0.00	4	0	0.00
myciel5.col	47	236	21.83	2	47	4	0.01	4	0	0.00	4	0	0.00
myciel6.col	95	755	16.91	2	95	4	0.01	4	0	0.01	4	0	0.01
myciel7.col	191	2,360	13.01	2	191	4	0.14	4	0	0.11	4	0	0.04
qg.order30.col	900	26,100	6.45	30	900	30	32.50	30	0	34.02	30	0	10.69
qg.order40.col	1,600	62,400	4.88	40	1,600	40	263.71	40	0	422.49	40	0	77.94
qg.order60.col	3,600	212,400	3.28	60	3,600	60	†	60	1,887	†	60	1,177	†
queen10_10.col	100	1,470	29.70	10	100	10	0.06	10	0	0.06	10	0	0.03
queen11_11.col	121	1,980	27.27	11	121	11	0.19	11	0	0.10	11	0	0.04
queen12_12.col	144	2,596	25.21	12	144	12	0.27	12	0	0.17	12	0	0.08
queen13_13.col	169	3,328	23.44	13	169	13	0.50	13	0	0.30	13	0	0.09
queen14_14.col	196	4,186	21.91	14	196	14	0.55	14	0	0.44	14	0	0.12
queen15_15.col	225	5,180	20.56	15	225	15	0.84	15	0	0.68	15	0	0.20

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
queen16_16.col	256	6,320	19.36	16	256	16	1.47	16	0	1.01	16	0	0.33
queen5_5.col	25	160	53.33	5	25	6	0.01	6	0	0.00	6	0	0.00
queen6_6.col	36	290	46.03	6	36	6	0.01	6	0	0.00	6	0	0.00
queen7_7.col	49	476	40.48	7	49	7	0.02	7	0	0.01	7	0	0.01
queen8_12.col	96	1,368	30.00	12	96	12	0.05	12	0	0.04	12	0	0.02
queen8_8.col	64	728	36.11	8	64	8	0.02	8	0	0.02	8	0	0.01
queen9_9.col	81	1,056	32.59	9	81	9	0.03	9	0	0.03	9	0	0.02
r1000.1.col	1,000	14,378	2.88	20	665	21	0.03	21	0	0.33	21	0	0.16
r1000.1c.col	1,000	485,090	97.12	91	1,000	33	†	33	945	†	33	945	†
r1000.5.col	1,000	238,267	47.70	234	984	112	†	133	478	†	134	477	†
r125.1.col	125	209	2.70	5	101	6	0.02	6	0	0.00	6	0	0.00
r125.1c.col	125	7,501	96.79	46	125	36	†	36	67	†	36	67	†
r125.5.col	125	3,838	49.52	36	119	37	0.86	37	0	0.44	37	0	0.41
r250.1.col	250	867	2.79	8	140	9	0.01	9	0	0.01	9	0	0.01
r250.1c.col	250	30,227	97.12	63	250	37	†	38	190	†	38	189	†
r250.5.col	250	14,849	47.71	65	237	66	20.76	66	0	6.57	66	0	4.00
school1.col	385	19,095	25.83	14	363	16	†	16	159	†	16	136	†
school1_nsh.col	352	14,612	23.65	14	332	16	†	16	64	†	16	62	†
wap01a.col	2,368	110,871	3.96	41	2,166	41	136.28	41	0	562.12	41	0	126.70
wap02a.col	2,464	111,742	3.68	40	2,362	41	137.58	41	83	†	41	0	140.20
wap03a.col	4,730	286,722	2.56	40	4,702	41	†	40	2,642	†	41	1,102	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
wap04a.col	5,231	294,902	2.16	40	5,205	41	†	40	3,018	†	41	1,562	†
wap05a.col	905	43,081	10.53	50	679	50	3.93	50	0	3.59	50	0	1.50
wap06a.col	947	43,571	9.73	40	834	40	39.00	40	0	33.71	40	0	10.96
wap07a.col	1,809	103,368	6.32	40	1,710	41	232.85	41	0	451.76	41	0	93.53
wap08a.col	1,870	104,176	5.96	40	1,763	40	254.22	40	0	473.67	40	0	99.41
will199GPIA.col	701	6,772	2.76	6	700	8	0.14	8	0	3.10	8	0	1.16
zeroin.i.1.col	211	4,100	18.51	49	79	50	0.48	50	0	0.14	50	0	0.18
zeroin.i.2.col	211	3,541	15.98	30	131	31	0.09	31	0	0.05	31	0	0.06
zeroin.i.3.col	206	3,540	16.77	30	131	31	0.09	31	0	0.05	31	0	0.06

Table A.10: RDS performance for 3-defective clique.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	19	†	19	97	†	21	79	†
brock200_2.clq	200	9,876	49.63	12	200	13	133.15	13	0	161.15	13	0	34.64
brock200_3.clq	200	12,048	60.54	15	200	16	†	15	41	†	16	6	†
brock200_4.clq	200	13,089	65.77	17	200	16	†	16	63	†	17	38	†
brock400_1.clq	400	59,723	74.84	27	400	19	†	19	293	†	20	282	†
brock400_2.clq	400	59,786	74.92	29	400	18	†	18	299	†	19	288	†
brock400_3.clq	400	59,681	74.79	31	400	20	†	20	292	†	21	280	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock400_4.clq	400	59,765	74.89	33	400	20	†	20	289	†	20	282	†
brock800_1.clq	800	207,505	64.93	23	800	16	†	16	651	†	17	631	†
brock800_2.clq	800	208,166	65.13	24	800	16	†	16	659	†	18	628	†
brock800_3.clq	800	207,333	64.87	25	800	16	†	16	662	†	17	632	†
brock800_4.clq	800	207,643	64.97	26	800	15	†	15	664	†	17	631	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.01	12	0	0.04	12	0	0.02
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.01	24	0	0.01	24	0	0.01
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.01	58	0	0.00	58	0	0.01
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.01	14	0	1.19	14	0	0.26
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.03	126	0	0.03	126	0	0.05
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.01	26	0	0.26	26	0	0.07
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.02	64	0	0.05	64	0	0.04
hamming10-2.clq	1,024	518,656	99.02	512	1,024	512	70.17	512	0	14.46	512	0	12.24
hamming10-4.clq	1,024	434,176	82.89	40	1,024	18	†	18	746	†	18	739	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.01	32	0	0.00	32	0	0.03
hamming6-4.clq	64	704	34.92	4	64	6	0.02	6	0	0.01	6	0	0.02
hamming8-2.clq	256	31,616	96.86	128	256	128	0.53	128	0	0.13	128	0	0.13
hamming8-4.clq	256	20,864	63.92	16	256	16	83.30	16	0	87.15	16	0	27.67
johnson16-2-4.clq	120	5,460	76.47	8	120	9	†	9	20	†	9	13	†
johnson32-2-4.clq	496	107,880	87.88	16	496	8	†	8	370	†	8	369	†
johnson8-2-4.clq	28	210	55.56	4	28	5	0.01	5	0	0.00	5	0	0.01

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
johnson8-4-4.clq	70	1,855	76.81	14	70	14	0.19	14	0	0.13	14	0	0.08
keller4.clq	171	9,435	64.91	11	171	14	580.25	14	0	570.33	14	0	162.00
keller5.clq	776	225,990	75.16	27	776	15	†	15	596	†	16	575	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	15	†	15	3,181	†	16	3,160	†
MANN_a27.clq	378	70,551	99.02	126	378	18	†	19	330	†	19	331	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	18	†	19	987	†	19	988	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	18	†	19	3,273	†	19	3,274	†
MANN_a9.clq	45	918	92.73	16	45	19	11.95	19	0	5.44	19	0	7.33
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	11	†	11	495	†	11	330	†
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	22	†	22	822	†	22	815	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	23	†	25	898	†	25	895	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	11	†	11	1,001	†	12	821	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	22	†	23	1,337	†	24	1,315	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	26	†	27	1,405	†	27	1,402	†
p_hat300-1.clq	300	10,933	24.38	8	300	10	24.68	10	0	24.48	10	0	5.73
p_hat300-2.clq	300	21,928	48.89	25	300	23	†	23	105	†	23	90	†
p_hat300-3.clq	300	33,390	74.45	36	300	24	†	24	198	†	27	181	†
p_hat500-1.clq	500	31,569	25.31	9	500	11	583.54	11	0	593.78	11	0	117.06
p_hat500-2.clq	500	62,946	50.46	36	500	28	†	28	293	†	29	270	†
p_hat500-3.clq	500	93,800	75.19	50	500	28	†	29	398	†	29	396	†
p_hat700-1.clq	700	60,999	24.93	11	700	11	†	11	202	†	13	0	334.24

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat700-2.clq	700	121,728	49.76	44	700	26	†	26	490	†	26	481	†
p_hat700-3.clq	700	183,010	74.81	62	700	25	†	25	586	†	29	575	†
san1000.clq	1,000	250,500	50.15	15	1,000	11	†	11	901	†	12	889	†
san200_0.7_1.clq	200	13,930	70.00	30	200	18	†	18	116	†	18	114	†
san200_0.7_2.clq	200	13,930	70.00	18	200	16	†	16	123	†	16	123	†
san200_0.9_1.clq	200	17,910	90.00	70	200	35	†	35	118	†	35	118	†
san200_0.9_2.clq	200	17,910	90.00	60	200	27	†	28	136	†	29	133	†
san200_0.9_3.clq	200	17,910	90.00	44	200	25	†	26	133	†	26	132	†
san400_0.5_1.clq	400	39,900	50.00	13	400	11	†	11	279	†	11	272	†
san400_0.7_1.clq	400	55,860	70.00	40	400	21	†	21	328	†	21	328	†
san400_0.7_2.clq	400	55,860	70.00	30	400	18	†	18	315	†	19	311	†
san400_0.7_3.clq	400	55,860	70.00	22	400	16	†	17	309	†	17	307	†
san400_0.9_1.clq	400	71,820	90.00	100	400	31	†	31	325	†	31	324	†
sanr200_0.7.clq	200	13,868	69.69	18	200	17	†	17	79	†	17	67	†
sanr200_0.9.clq	200	17,863	89.76	42	200	27	†	27	132	†	27	131	†
sanr400_0.5.clq	400	39,984	50.11	13	400	13	†	13	176	†	14	128	†
sanr400_0.7.clq	400	55,869	70.01	21	400	18	†	18	278	†	19	261	†
adjnoun.graph	112	425	6.84	5	112	7	0.02	7	0	0.03	7	0	0.05
as-22july06.graph	22,963	48,436	0.02	17	204	19	0.23	19	0	0.15	19	0	0.08
astro-ph.graph	16,706	121,251	0.09	57	165	57	0.08	57	0	0.12	57	0	0.06
caidaRouterLevel.graph	192,244	609,066	0.00	17	5,417	20	64.57	6	4,251	†	7	3,565	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
celegans_metabolic.graph	453	2,025	1.98	9	240	11	0.08	11	0	0.18	11	0	0.05
celegansneural.graph	297	2,148	4.89	8	274	10	0.25	10	0	0.63	10	0	0.17
chesapeake.graph	39	170	22.94	5	39	7	0.02	7	0	0.00	7	0	0.01
cnr-2000.graph	325,557	2,738,969	0.01	84	170	86	0.01	86	0	1.56	86	0	0.25
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.02	87	0	0.00	87	0	0.04
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.02	115	0	0.00	115	0	0.05
cond-mat.graph	16,726	47,594	0.03	18	98	18	0.02	18	0	0.01	18	0	0.01
cond-mat-2003.graph	31,163	120,029	0.03	25	77	26	0.05	26	0	0.05	26	0	0.03
cond-mat-2005.graph	40,421	175,691	0.02	30	57	30	0.02	30	0	0.00	30	0	0.01
dolphins.graph	62	159	8.41	5	62	6	0.01	6	0	0.00	6	0	0.00
email.graph	1,133	5,451	0.85	12	349	12	0.28	12	0	3.63	12	0	0.60
football.graph	115	613	9.35	9	115	9	0.01	9	0	0.03	9	0	0.01
hep-th.graph	8,361	15,751	0.05	24	24	24	0.01	24	0	0.00	24	0	0.01
jazz.graph	198	2,742	14.06	30	30	30	0.02	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	34	6	0.01	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	38	11	0.01	11	0	0.00	11	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.01	97	0	0.00	97	0	0.04
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.03
PGPgiantcompo.graph	10,680	24,316	0.04	25	171	28	0.14	28	0	0.11	28	0	0.07
polblogs.graph	1,490	16,715	1.51	20	517	22	20.83	22	0	24.98	22	0	8.95
polbooks.graph	105	441	8.08	6	105	8	0.02	8	0	0.03	8	0	0.06

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
power.graph	4,941	6,594	0.05	6	3,353	7	0.06	7	2,284	†	7	1,748	†
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	2,016	16	0.05	16	849	†	16	156	†
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	534	19	0.01	19	0	22.80	19	0	4.02
Cit-HepTh.txt	27,769	352,285	0.09	23	8,168	25	†	13	6,902	†	19	5,537	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,252	18	86.10	7	1,187	†	18	0	373.22
Slashdot0811.txt	77,360	469,180	0.02	26	6,974	7	†	5	5,931	†	6	5,414	†
Slashdot0902.txt	82,168	504,230	0.02	27	6,978	8	†	5	5,907	†	5	5,432	†
soc-Epinions1.txt	75,879	405,740	0.01	23	5,719	22	†	18	4,430	†	22	3,517	†
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	202	72.29	202	0	91.92	202	0	14.96
web-Google.txt	875,713	4,322,051	0.00	44	223	45	†	46	0	0.36	46	0	0.27
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	152	†	152	772	†	152	491	†
web-Stanford.txt	281,903	1,992,636	0.01	61	1,499	59	†	43	623	†	46	554	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,520	17	†	15	1,247	†	19	447	†
1-FullIns_3.col	30	100	22.99	3	30	5	0.01	5	0	0.00	5	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	6	0.02	6	0	0.03	6	0	0.02
1-FullIns_5.col	282	3,247	8.20	3	282	6	0.45	6	0	1.60	6	0	0.40
1-Insertions_4.col	67	232	10.49	2	67	4	0.02	4	0	0.01	4	0	0.01
1-Insertions_5.col	202	1,227	6.04	2	202	4	0.84	4	0	0.53	4	0	0.16
1-Insertions_6.col	607	6,337	3.45	2	607	4	65.12	4	0	41.39	4	0	6.51
2-FullIns_3.col	52	201	15.16	4	52	6	0.01	6	0	0.00	6	0	0.01
2-FullIns_4.col	212	1,621	7.25	4	212	6	0.09	6	0	0.42	6	0	0.13

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
2-FullIns_5.col	852	12,201	3.37	4	852	7	7.46	7	0	102.28	7	0	20.69
2-Insertions_3.col	37	72	10.81	2	37	4	0.01	4	0	0.00	4	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	4	0.25	4	0	0.16	4	0	0.05
2-Insertions_5.col	597	3,936	2.21	2	597	4	59.11	4	0	37.51	4	0	5.88
3-FullIns_3.col	80	346	10.95	5	80	7	0.02	7	0	0.01	7	0	0.02
3-FullIns_4.col	405	3,524	4.31	5	405	7	0.30	7	0	5.04	7	0	0.97
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	8	88.13	6	967	†	6	471	†
3-Insertions_3.col	56	110	7.14	2	56	4	0.01	4	0	0.00	4	0	0.00
3-Insertions_4.col	281	1,046	2.66	2	281	4	2.95	4	0	2.18	4	0	0.35
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	4	†	4	417	†	4	0	384.41
4-FullIns_3.col	114	541	8.40	6	114	8	0.02	8	0	0.06	8	0	0.02
4-FullIns_4.col	690	6,650	2.80	6	690	8	1.09	8	0	42.53	8	0	5.89
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	6	†	6	3,108	†	6	2,600	†
4-Insertions_3.col	79	156	5.06	2	79	4	0.02	4	0	0.01	4	0	0.03
4-Insertions_4.col	475	1,795	1.59	2	475	4	23.04	4	0	20.37	4	0	2.50
5-FullIns_3.col	154	792	6.72	7	154	9	0.01	9	0	0.16	9	0	0.04
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	9	3.67	9	0	425.37	9	0	68.73
abb313GPIA.col	1,557	53,356	4.41	8	1,555	11	†	11	803	†	11	571	†
anna.col	138	493	5.22	11	24	12	0.02	12	0	0.00	12	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	5	0.14	5	0	44.51	5	0	7.12
ash608GPIA.col	1,216	7,844	1.06	3	1,216	4	†	4	150	†	4	0	208.36

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
ash958GPIA.col	1,916	12,506	0.68	3	1,916	4	†	4	875	†	4	361	†
C2000.5.col	2,000	999,836	50.02	16	2,000	13	†	13	1,762	†	14	1,701	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	13	†	13	3,765	†	13	3,721	†
david.col	87	406	10.85	11	36	12	0.01	12	0	0.00	12	0	0.02
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	7	†	7	43	†	7	0	121.13
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	13	†	13	765	†	14	703	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	27	†	28	934	†	29	932	†
DSJC125.1.col	125	736	9.50	4	125	6	0.03	6	0	0.07	6	0	0.03
DSJC125.5.col	125	3,891	50.21	10	125	12	11.47	12	0	11.78	12	0	3.15
DSJC125.9.col	125	6,961	89.82	34	125	27	†	27	59	†	27	58	†
DSJC250.1.col	250	3,218	10.34	4	250	6	1.30	6	0	1.32	6	0	0.28
DSJC250.5.col	250	15,668	50.34	12	250	13	†	13	24	†	13	0	257.25
DSJC250.9.col	250	27,897	89.63	43	250	29	†	29	176	†	29	174	†
DSJC500.1.col	500	12,458	9.99	5	500	6	38.91	6	0	39.40	6	0	5.49
DSJC500.5.col	500	62,624	50.20	13	500	13	†	13	269	†	14	206	†
DSJC500.9.col	500	112,437	90.13	56	500	26	†	26	433	†	26	432	†
DSJR500.1.col	500	3,555	2.85	11	441	14	0.01	14	0	1.62	14	0	0.31
DSJR500.1c.col	500	121,275	97.21	83	500	34	†	36	446	†	36	445	†
DSJR500.5.col	500	58,862	47.18	122	492	69	†	77	199	†	77	188	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	13	†	13	752	†	13	712	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	13	†	13	762	†	13	712	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	13	†	13	759	†	13	714	†
flat300_20_0.col	300	21,375	47.66	11	300	12	†	12	63	†	13	0	426.99
flat300_26_0.col	300	21,633	48.23	11	300	13	†	13	54	†	13	0	508.39
flat300_28_0.col	300	21,695	48.37	12	300	12	†	12	67	†	13	0	558.67
fpsol2.i.1.col	496	11,654	9.49	65	91	66	0.16	66	0	0.03	66	0	0.04
fpsol2.i.2.col	451	8,691	8.57	30	238	31	0.42	31	0	0.22	31	0	0.10
fpsol2.i.3.col	425	8,688	9.64	30	238	31	0.44	31	0	0.21	31	0	0.11
games120.col	120	638	8.94	9	120	9	0.02	9	0	0.02	9	0	0.01
homer.col	561	1,628	1.04	13	68	13	0.02	13	0	0.00	13	0	0.00
huck.col	74	301	11.14	11	42	11	0.01	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	150	56	5.04	56	0	1.16	56	0	1.11
inithx.i.2.col	645	13,979	6.73	31	338	32	4.31	32	0	2.11	32	0	1.78
inithx.i.3.col	621	13,969	7.26	31	335	32	2.18	32	0	1.06	32	0	0.67
jean.col	80	254	8.04	10	38	11	0.01	11	0	0.00	11	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	788	†	90	786	†
le450_15a.col	450	8,168	8.09	15	420	15	1.14	15	0	1.58	15	0	0.55
le450_15b.col	450	8,169	8.09	15	427	15	2.06	15	0	2.99	15	0	0.64
le450_15c.col	450	16,680	16.51	15	450	16	5.55	16	0	5.78	16	0	1.27
le450_15d.col	450	16,750	16.58	15	450	15	11.72	15	0	13.22	15	0	2.66
le450_25a.col	450	8,260	8.18	25	289	25	0.14	25	0	0.20	25	0	0.07
le450_25b.col	450	8,263	8.18	25	314	25	0.31	25	0	0.33	25	0	0.10

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
le450_25c.col	450	17,343	17.17	25	439	25	4.85	25	0	5.57	25	0	1.41
le450_25d.col	450	17,425	17.25	25	441	25	2.53	25	0	3.10	25	0	0.84
le450_5a.col	450	5,714	5.66	5	450	7	2.29	7	0	7.24	7	0	1.21
le450_5b.col	450	5,734	5.68	5	450	7	2.59	7	0	9.48	7	0	1.35
le450_5c.col	450	9,803	9.70	5	450	7	17.05	7	0	18.68	7	0	2.65
le450_5d.col	450	9,757	9.66	5	450	7	14.77	7	0	16.82	7	0	2.42
miles1000.col	128	3,216	39.57	42	62	44	0.06	44	0	0.03	44	0	0.02
miles1500.col	128	5,198	63.95	73	86	74	1.59	74	0	0.40	74	0	0.31
miles250.col	128	387	4.76	8	83	9	0.01	9	0	0.00	9	0	0.01
miles500.col	128	1,170	14.40	20	36	22	0.01	22	0	0.00	22	0	0.00
miles750.col	128	2,113	26.00	31	43	33	0.02	33	0	0.00	33	0	0.00
mug100_1.col	100	166	3.35	3	100	4	0.05	4	0	0.06	4	0	0.01
mug100_25.col	100	166	3.35	3	100	4	0.05	4	0	0.06	4	0	0.01
mug88_1.col	88	146	3.81	3	88	4	0.03	4	0	0.02	4	0	0.01
mug88_25.col	88	146	3.81	3	88	4	0.03	4	0	0.03	4	0	0.01
mulsol.i.1.col	197	3,925	20.33	49	63	51	0.05	51	0	0.01	51	0	0.01
mulsol.i.2.col	188	3,885	22.10	31	122	32	0.11	32	0	0.08	32	0	0.03
mulsol.i.3.col	184	3,916	23.26	31	123	32	0.13	32	0	0.08	32	0	0.04
mulsol.i.4.col	185	3,946	23.18	31	124	32	0.13	32	0	0.06	32	0	0.06
mulsol.i.5.col	186	3,973	23.09	31	125	32	0.11	32	0	0.05	32	0	0.04
myciel3.col	11	20	36.36	2	11	4	0.01	4	0	0.00	4	0	0.00

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
myciel4.col	23	71	28.06	2	23	4	0.01	4	0	0.00	4	0	0.00
myciel5.col	47	236	21.83	2	47	4	0.02	4	0	0.01	4	0	0.00
myciel6.col	95	755	16.91	2	95	4	0.22	4	0	0.08	4	0	0.03
myciel7.col	191	2,360	13.01	2	191	4	2.96	4	0	1.30	4	0	0.32
qg.order30.col	900	26,100	6.45	30	900	30	575.77	30	0	557.22	30	0	94.73
qg.order40.col	1,600	62,400	4.88	40	1,600	40	†	40	728	†	40	259	†
qg.order60.col	3,600	212,400	3.28	60	3,600	60	†	60	2,824	†	60	2,401	†
queen10_10.col	100	1,470	29.70	10	100	10	0.28	10	0	0.24	10	0	0.09
queen11_11.col	121	1,980	27.27	11	121	11	0.53	11	0	0.48	11	0	0.15
queen12_12.col	144	2,596	25.21	12	144	12	0.98	12	0	0.88	12	0	0.23
queen13_13.col	169	3,328	23.44	13	169	13	1.76	13	0	1.70	13	0	0.39
queen14_14.col	196	4,186	21.91	14	196	14	2.98	14	0	2.95	14	0	0.57
queen15_15.col	225	5,180	20.56	15	225	15	4.95	15	0	4.26	15	0	1.01
queen16_16.col	256	6,320	19.36	16	256	16	8.16	16	0	7.81	16	0	1.75
queen5_5.col	25	160	53.33	5	25	6	0.01	6	0	0.00	6	0	0.00
queen6_6.col	36	290	46.03	6	36	7	0.02	7	0	0.01	7	0	0.00
queen7_7.col	49	476	40.48	7	49	7	0.03	7	0	0.02	7	0	0.01
queen8_12.col	96	1,368	30.00	12	96	12	0.25	12	0	0.19	12	0	0.07
queen8_8.col	64	728	36.11	8	64	8	0.06	8	0	0.05	8	0	0.02
queen9_9.col	81	1,056	32.59	9	81	9	0.14	9	0	0.12	9	0	0.05
r1000.1.col	1,000	14,378	2.88	20	844	22	0.08	22	0	16.96	22	0	3.49

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
r1000.1c.col	1,000	485,090	97.12	91	1,000	33	†	34	946	†	34	946	†
r1000.5.col	1,000	238,267	47.70	234	985	74	†	76	705	†	86	667	†
r125.1.col	125	209	2.70	5	122	6	0.01	6	0	0.03	6	0	0.04
r125.1c.col	125	7,501	96.79	46	125	37	†	37	68	†	37	68	†
r125.5.col	125	3,838	49.52	36	120	37	4.12	37	0	1.83	37	0	0.98
r250.1.col	250	867	2.79	8	203	9	0.01	9	0	0.33	9	0	0.11
r250.1c.col	250	30,227	97.12	63	250	37	†	38	192	†	39	191	†
r250.5.col	250	14,849	47.71	65	238	66	238.11	66	0	64.32	66	0	37.84
school1.col	385	19,095	25.83	14	363	17	†	17	201	†	17	186	†
school1_nsh.col	352	14,612	23.65	14	332	17	†	17	116	†	17	72	†
wap01a.col	2,368	110,871	3.96	41	2,281	41	†	41	1,191	†	41	691	†
wap02a.col	2,464	111,742	3.68	40	2,372	41	†	40	1,259	†	41	713	†
wap03a.col	4,730	286,722	2.56	40	4,702	41	†	40	3,504	†	41	2,922	†
wap04a.col	5,231	294,902	2.16	40	5,207	40	†	40	3,707	†	40	3,416	†
wap05a.col	905	43,081	10.53	50	685	50	27.69	50	0	26.64	50	0	8.89
wap06a.col	947	43,571	9.73	40	846	40	313.44	40	0	345.56	40	0	82.47
wap07a.col	1,809	103,368	6.32	40	1,719	42	†	42	617	†	42	204	†
wap08a.col	1,870	104,176	5.96	40	1,773	40	†	40	722	†	40	319	†
will199GPIA.col	701	6,772	2.76	6	701	8	0.55	8	0	84.56	8	0	14.24
zeroin.i.1.col	211	4,100	18.51	49	79	50	0.95	50	0	0.27	50	0	0.32
zeroin.i.2.col	211	3,541	15.98	30	136	31	0.31	31	0	0.16	31	0	0.14

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
zeroin.i.3.col	206	3,540	16.77	30	136	31	0.31	31	0	0.16	31	0	0.15

Table A.11: RDS performance for 4-defective clique.

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
brock200_1.clq	200	14,834	74.54	21	200	19	†	19	109	†	19	103	†
brock200_2.clq	200	9,876	49.63	12	200	13	†	13	14	†	13	0	249.13
brock200_3.clq	200	12,048	60.54	15	200	15	†	15	72	†	15	47	†
brock200_4.clq	200	13,089	65.77	17	200	16	†	16	81	†	16	71	†
brock400_1.clq	400	59,723	74.84	27	400	19	†	19	307	†	19	302	†
brock400_2.clq	400	59,786	74.92	29	400	18	†	18	310	†	19	300	†
brock400_3.clq	400	59,681	74.79	31	400	19	†	19	311	†	21	294	†
brock400_4.clq	400	59,765	74.89	33	400	18	†	18	311	†	19	302	†
brock800_1.clq	800	207,505	64.93	23	800	15	†	16	678	†	17	655	†
brock800_2.clq	800	208,166	65.13	24	800	17	†	17	676	†	17	662	†
brock800_3.clq	800	207,333	64.87	25	800	16	†	16	682	†	17	664	†
brock800_4.clq	800	207,643	64.97	26	800	16	†	16	676	†	16	663	†
c-fat200-1.clq	200	1,534	7.71	12	200	12	0.01	12	0	0.08	12	0	0.05
c-fat200-2.clq	200	3,235	16.26	24	200	24	0.01	24	0	0.03	24	0	0.03
c-fat200-5.clq	200	8,473	42.58	58	200	58	0.02	58	0	0.01	58	0	0.02

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
c-fat500-1.clq	500	4,459	3.57	14	500	14	0.01	14	0	1.65	14	0	0.50
c-fat500-10.clq	500	46,627	37.38	126	500	126	0.06	126	0	0.08	126	0	0.12
c-fat500-2.clq	500	9,139	7.33	26	500	26	0.01	26	0	0.39	26	0	0.22
c-fat500-5.clq	500	23,191	18.59	64	500	64	0.02	64	0	0.12	64	0	0.14
hamming10-2.clq	1,024	518,656	99.02	512	1,024	512	273.07	512	0	54.94	512	0	47.48
hamming10-4.clq	1,024	434,176	82.89	40	1,024	15	†	15	801	†	18	757	†
hamming6-2.clq	64	1,824	90.48	32	64	32	0.02	32	0	0.00	32	0	0.01
hamming6-4.clq	64	704	34.92	4	64	6	0.09	6	0	0.05	6	0	0.02
hamming8-2.clq	256	31,616	96.86	128	256	128	1.67	128	0	0.36	128	0	0.34
hamming8-4.clq	256	20,864	63.92	16	256	15	†	14	57	†	17	0	230.86
johnson16-2-4.clq	120	5,460	76.47	8	120	10	†	10	29	†	10	23	†
johnson32-2-4.clq	496	107,880	87.88	16	496	9	†	9	374	†	9	373	†
johnson8-2-4.clq	28	210	55.56	4	28	6	0.02	6	0	0.00	6	0	0.01
johnson8-4-4.clq	70	1,855	76.81	14	70	15	0.87	15	0	0.91	15	0	0.50
keller4.clq	171	9,435	64.91	11	171	15	†	15	3	†	15	0	330.10
keller5.clq	776	225,990	75.16	27	776	15	†	15	612	†	15	606	†
keller6.clq	3,361	4,619,898	81.82	59	3,361	15	†	15	3,197	†	15	3,192	†
MANN_a27.clq	378	70,551	99.02	126	378	18	†	19	334	†	19	333	†
MANN_a45.clq	1,035	533,115	99.63	345	1,035	18	†	19	991	†	19	991	†
MANN_a81.clq	3,321	5,506,380	99.88	1,100	3,321	18	†	19	3,277	†	19	3,277	†
MANN_a9.clq	45	918	92.73	16	45	20	41.56	20	0	18.16	20	0	23.84

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
p_hat1000-1.clq	1,000	122,253	24.48	10	1,000	11	†	11	632	†	12	512	†
p_hat1000-2.clq	1,000	244,799	49.01	46	1,000	22	†	22	838	†	23	823	†
p_hat1000-3.clq	1,000	371,746	74.42	68	1,000	23	†	23	909	†	24	906	†
p_hat1500-1.clq	1,500	284,923	25.34	12	1,500	11	†	11	1,113	†	12	998	†
p_hat1500-2.clq	1,500	568,960	50.61	65	1,500	23	†	23	1,352	†	23	1,343	†
p_hat1500-3.clq	1,500	847,244	75.36	94	1,500	25	†	26	1,414	†	26	1,412	†
p_hat300-1.clq	300	10,933	24.38	8	300	10	164.80	10	0	158.42	10	0	36.97
p_hat300-2.clq	300	21,928	48.89	25	300	22	†	22	145	†	22	134	†
p_hat300-3.clq	300	33,390	74.45	36	300	24	†	25	206	†	25	203	†
p_hat500-1.clq	500	31,569	25.31	9	500	11	†	11	131	†	11	36	†
p_hat500-2.clq	500	62,946	50.46	36	500	25	†	26	335	†	27	320	†
p_hat500-3.clq	500	93,800	75.19	50	500	29	†	29	403	†	30	398	†
p_hat700-1.clq	700	60,999	24.93	11	700	11	†	11	324	†	12	193	†
p_hat700-2.clq	700	121,728	49.76	44	700	23	†	23	527	†	26	496	†
p_hat700-3.clq	700	183,010	74.81	62	700	24	†	24	600	†	25	590	†
san1000.clq	1,000	250,500	50.15	15	1,000	12	†	12	908	†	12	906	†
san200_0.7_1.clq	200	13,930	70.00	30	200	19	†	19	119	†	19	118	†
san200_0.7_2.clq	200	13,930	70.00	18	200	16	†	16	130	†	16	129	†
san200_0.9_1.clq	200	17,910	90.00	70	200	35	†	36	120	†	36	119	†
san200_0.9_2.clq	200	17,910	90.00	60	200	28	†	28	140	†	29	138	†
san200_0.9_3.clq	200	17,910	90.00	44	200	26	†	26	137	†	26	137	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
san400_0.5_1.clq	400	39,900	50.00	13	400	11	†	11	301	†	12	288	†
san400_0.7_1.clq	400	55,860	70.00	40	400	22	†	22	329	†	22	329	†
san400_0.7_2.clq	400	55,860	70.00	30	400	19	†	19	317	†	19	317	†
san400_0.7_3.clq	400	55,860	70.00	22	400	17	†	17	313	†	17	311	†
san400_0.9_1.clq	400	71,820	90.00	100	400	31	†	32	328	†	32	326	†
sanr200_0.7.clq	200	13,868	69.69	18	200	17	†	18	92	†	18	81	†
sanr200_0.9.clq	200	17,863	89.76	42	200	27	†	28	135	†	28	134	†
sanr400_0.5.clq	400	39,984	50.11	13	400	13	†	13	221	†	13	186	†
sanr400_0.7.clq	400	55,869	70.01	21	400	18	†	18	296	†	18	285	†
adjnoun.graph	112	425	6.84	5	112	7	0.02	7	0	0.05	7	0	0.03
as-22july06.graph	22,963	48,436	0.02	17	232	19	0.53	19	0	0.33	19	0	0.18
astro-ph.graph	16,706	121,251	0.09	57	165	57	0.73	57	0	1.04	57	0	0.36
caidaRouterLevel.graph	192,244	609,066	0.00	17	6,447	20	270.51	7	5,242	†	7	4,740	†
celegans_metabolic.graph	453	2,025	1.98	9	313	11	0.27	11	0	0.77	11	0	0.24
celegansneural.graph	297	2,148	4.89	8	278	10	0.52	10	0	1.85	10	0	0.56
chesapeake.graph	39	170	22.94	5	39	7	0.02	7	0	0.00	7	0	0.00
cnr-2000.graph	325,557	2,738,969	0.01	84	286	80	†	80	183	†	80	173	†
coAuthorsCiteseer.graph	227,320	814,134	0.00	87	87	87	0.01	87	0	0.00	87	0	0.01
coAuthorsDBLP.graph	299,067	977,676	0.00	115	115	115	0.03	115	0	0.00	115	0	0.01
cond-mat.graph	16,726	47,594	0.03	18	164	18	0.02	18	0	0.19	18	0	0.05
cond-mat-2003.graph	31,163	120,029	0.03	25	77	26	0.05	26	0	0.16	26	0	0.05

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
cond-mat-2005.graph	40,421	175,691	0.02	30	83	30	0.01	30	0	0.11	30	0	0.05
dolphins.graph	62	159	8.41	5	62	7	0.01	7	0	0.00	7	0	0.00
email.graph	1,133	5,451	0.85	12	434	13	0.75	13	0	13.14	13	0	2.42
football.graph	115	613	9.35	9	115	9	0.02	9	0	0.04	9	0	0.02
hep-th.graph	8,361	15,751	0.05	24	24	24	0.02	24	0	0.00	24	0	0.00
jazz.graph	198	2,742	14.06	30	30	30	0.01	30	0	0.00	30	0	0.00
karate.graph	34	78	13.90	5	34	6	0.01	6	0	0.00	6	0	0.00
lesmis.graph	77	254	8.68	10	38	12	0.02	12	0	0.00	12	0	0.00
memplus.graph	17,758	54,196	0.03	97	97	97	0.01	97	0	0.00	97	0	0.01
netscience.graph	1,589	2,742	0.22	20	20	20	0.01	20	0	0.00	20	0	0.00
PGPgiantcompo.graph	10,680	24,316	0.04	25	172	28	0.19	28	0	0.15	28	0	0.12
polblogs.graph	1,490	16,715	1.51	20	541	23	69.00	23	0	111.89	23	0	27.57
polbooks.graph	105	441	8.08	6	105	8	0.01	8	0	0.16	8	0	0.03
power.graph	4,941	6,594	0.05	6	4,941	7	0.14	7	3,794	†	7	3,265	†
rgg_n_2_17_s0.graph	131,072	728,474	0.01	15	6,441	16	0.34	15	5,542	†	15	5,029	†
rgg_n_2_19_s0.graph	524,288	3,269,220	0.00	18	1,995	20	0.03	18	1,033	†	18	578	†
Cit-HepTh.txt	27,769	352,285	0.09	23	8,595	25	†	13	7,313	†	17	6,463	†
Email-EuAll.txt	265,009	364,481	0.00	16	2,498	18	489.92	8	1,297	†	10	539	†
Slashdot0811.txt	77,360	469,180	0.02	26	7,397	8	†	5	6,348	†	6	5,675	†
Slashdot0902.txt	82,168	504,230	0.02	27	7,368	6	†	5	6,335	†	6	5,620	†
soc-Epinions1.txt	75,879	405,740	0.01	23	6,010	21	†	19	4,709	†	22	3,839	†

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
web-BerkStan.txt	685,230	6,649,470	0.00	201	392	162	†	201	57	†	202	0	518.95
web-Google.txt	875,713	4,322,051	0.00	44	223	45	†	47	0	0.54	47	0	0.48
web-NotreDame.txt	325,729	1,090,108	0.00	155	1,367	150	†	150	1,077	†	150	939	†
web-Stanford.txt	281,903	1,992,636	0.01	61	1,595	37	†	43	688	†	43	658	†
Wiki-Vote.txt	7,115	100,762	0.40	17	2,604	15	†	14	1,639	†	15	1,337	†
1-FullIns_3.col	30	100	22.99	3	30	6	0.01	6	0	0.00	6	0	0.00
1-FullIns_4.col	93	593	13.86	3	93	6	0.11	6	0	0.07	6	0	0.03
1-FullIns_5.col	282	3,247	8.20	3	282	6	9.17	6	0	5.49	6	0	1.59
1-Insertions_4.col	67	232	10.49	2	67	5	0.03	5	0	0.02	5	0	0.01
1-Insertions_5.col	202	1,227	6.04	2	202	5	1.67	5	0	1.07	5	0	0.34
1-Insertions_6.col	607	6,337	3.45	2	607	5	135.30	5	0	88.84	5	0	17.73
2-FullIns_3.col	52	201	15.16	4	52	6	0.02	6	0	0.01	6	0	0.01
2-FullIns_4.col	212	1,621	7.25	4	212	7	0.09	7	0	0.96	7	0	0.35
2-FullIns_5.col	852	12,201	3.37	4	852	7	9.67	7	0	240.45	7	0	48.38
2-Insertions_3.col	37	72	10.81	2	37	4	0.02	4	0	0.00	4	0	0.00
2-Insertions_4.col	149	541	4.91	2	149	5	0.36	5	0	0.23	5	0	0.09
2-Insertions_5.col	597	3,936	2.21	2	597	5	82.18	5	0	53.35	5	0	10.43
3-FullIns_3.col	80	346	10.95	5	80	7	0.01	7	0	0.03	7	0	0.02
3-FullIns_4.col	405	3,524	4.31	5	405	8	0.39	8	0	10.60	8	0	2.73
3-FullIns_5.col	2,030	33,751	1.64	5	2,030	8	103.74	6	1,026	†	6	582	†
3-Insertions_3.col	56	110	7.14	2	56	4	0.02	4	0	0.01	4	0	0.01

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
3-Insertions_4.col	281	1,046	2.66	2	281	5	3.73	5	0	2.77	5	0	0.59
3-Insertions_5.col	1,406	9,695	0.98	2	1,406	5	†	5	416	†	5	0	469.70
4-FullIns_3.col	114	541	8.40	6	114	8	0.02	8	0	0.14	8	0	0.05
4-FullIns_4.col	690	6,650	2.80	6	690	9	1.48	9	0	105.75	9	0	17.13
4-FullIns_5.col	4,146	77,305	0.90	6	4,146	7	†	7	3,133	†	7	2,624	†
4-Insertions_3.col	79	156	5.06	2	79	4	0.03	4	0	0.02	4	0	0.01
4-Insertions_4.col	475	1,795	1.59	2	475	5	26.30	5	0	21.30	5	0	3.48
5-FullIns_3.col	154	792	6.72	7	154	9	0.02	9	0	0.41	9	0	0.10
5-FullIns_4.col	1,085	11,395	1.94	7	1,085	10	4.45	7	150	†	10	0	99.58
abb313GPIA.col	1,557	53,356	4.41	8	1,555	12	†	12	1,133	†	12	793	†
anna.col	138	493	5.22	11	44	12	0.02	12	0	0.00	12	0	0.00
ash331GPIA.col	662	4,181	1.91	3	662	5	128.70	5	0	86.89	5	0	15.94
ash608GPIA.col	1,216	7,844	1.06	3	1,216	5	†	5	205	†	5	0	234.51
ash958GPIA.col	1,916	12,506	0.68	3	1,916	5	†	5	937	†	5	418	†
C2000.5.col	2,000	999,836	50.02	16	2,000	13	†	13	1,808	†	13	1,776	†
C4000.5.col	4,000	4,000,268	50.02	18	4,000	13	†	13	3,811	†	13	3,778	†
david.col	87	406	10.85	11	44	12	0.01	12	0	0.00	12	0	0.00
DSJC1000.1.col	1,000	49,629	9.94	6	1,000	7	†	7	307	†	7	51	†
DSJC1000.5.col	1,000	249,826	50.02	15	1,000	13	†	13	811	†	13	782	†
DSJC1000.9.col	1,000	449,449	89.98	68	1,000	28	†	28	938	†	28	937	†
DSJC125.1.col	125	736	9.50	4	125	6	0.30	6	0	0.19	6	0	0.08

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
DSJC125.5.col	125	3,891	50.21	10	125	12	41.62	12	0	42.90	12	0	12.55
DSJC125.9.col	125	6,961	89.82	34	125	26	†	28	61	†	28	61	†
DSJC250.1.col	250	3,218	10.34	4	250	6	12.56	6	0	5.48	6	0	1.41
DSJC250.5.col	250	15,668	50.34	12	250	13	†	13	79	†	13	34	†
DSJC250.9.col	250	27,897	89.63	43	250	29	†	29	181	†	29	180	†
DSJC500.1.col	500	12,458	9.99	5	500	7	129.50	7	0	131.83	7	0	22.75
DSJC500.5.col	500	62,624	50.20	13	500	13	†	13	317	†	14	264	†
DSJC500.9.col	500	112,437	90.13	56	500	26	†	26	437	†	26	435	†
DSJR500.1.col	500	3,555	2.85	11	489	15	0.02	15	0	2.49	15	0	0.64
DSJR500.1c.col	500	121,275	97.21	83	500	34	†	35	448	†	36	447	†
DSJR500.5.col	500	58,862	47.18	122	492	62	†	63	243	†	69	229	†
flat1000_50_0.col	1,000	245,000	49.05	15	1,000	13	†	13	796	†	13	766	†
flat1000_60_0.col	1,000	245,830	49.22	15	1,000	13	†	13	814	†	13	777	†
flat1000_76_0.col	1,000	246,708	49.39	15	1,000	13	†	13	804	†	13	771	†
flat300_20_0.col	300	21,375	47.66	11	300	12	†	12	109	†	13	54	†
flat300_26_0.col	300	21,633	48.23	11	300	13	†	13	98	†	13	62	†
flat300_28_0.col	300	21,695	48.37	12	300	13	†	13	96	†	13	64	†
fpsol2.i.1.col	496	11,654	9.49	65	120	66	6.37	66	0	1.26	66	0	1.19
fpsol2.i.2.col	451	8,691	8.57	30	260	31	2.00	31	0	0.97	31	0	0.49
fpsol2.i.3.col	425	8,688	9.64	30	260	31	2.00	31	0	0.96	31	0	0.44
games120.col	120	638	8.94	9	120	9	0.01	9	0	0.03	9	0	0.02

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
homer.col	561	1,628	1.04	13	98	13	0.02	13	0	0.02	13	0	0.01
huck.col	74	301	11.14	11	45	11	0.02	11	0	0.00	11	0	0.00
inithx.i.1.col	864	18,707	5.02	54	158	56	13.81	56	0	5.16	56	0	5.03
inithx.i.2.col	645	13,979	6.73	31	396	32	17.89	32	0	7.97	32	0	5.57
inithx.i.3.col	621	13,969	7.26	31	396	32	17.18	32	0	7.81	32	0	5.36
jean.col	80	254	8.04	10	38	12	0.01	12	0	0.00	12	0	0.00
latin_square_10.col	900	307,350	75.97	90	900	90	†	90	790	†	90	788	†
le450_15a.col	450	8,168	8.09	15	427	15	3.06	15	0	4.40	15	0	1.10
le450_15b.col	450	8,169	8.09	15	429	15	7.10	15	0	10.45	15	0	2.27
le450_15c.col	450	16,680	16.51	15	450	16	34.76	16	0	40.67	16	0	7.38
le450_15d.col	450	16,750	16.58	15	450	16	52.54	16	0	66.62	16	0	11.85
le450_25a.col	450	8,260	8.18	25	297	25	0.44	25	0	0.64	25	0	0.19
le450_25b.col	450	8,263	8.18	25	320	25	1.01	25	0	0.96	25	0	0.35
le450_25c.col	450	17,343	17.17	25	442	25	21.20	25	0	26.53	25	0	5.23
le450_25d.col	450	17,425	17.25	25	442	25	10.25	25	0	14.19	25	0	3.08
le450_5a.col	450	5,714	5.66	5	450	7	6.68	7	0	23.94	7	0	4.54
le450_5b.col	450	5,734	5.68	5	450	7	6.13	7	0	25.21	7	0	4.48
le450_5c.col	450	9,803	9.70	5	450	8	66.85	8	0	80.34	8	0	13.58
le450_5d.col	450	9,757	9.66	5	450	7	80.40	7	0	90.15	7	0	15.28
miles1000.col	128	3,216	39.57	42	81	44	0.06	44	0	0.02	44	0	0.02
miles1500.col	128	5,198	63.95	73	88	74	20.61	74	0	5.06	74	0	3.66

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
miles250.col	128	387	4.76	8	102	10	0.01	10	0	0.01	10	0	0.01
miles500.col	128	1,170	14.40	20	36	22	0.01	22	0	0.00	22	0	0.00
miles750.col	128	2,113	26.00	31	43	33	0.01	33	0	0.00	33	0	0.01
mug100_1.col	100	166	3.35	3	100	5	0.06	5	0	0.06	5	0	0.02
mug100_25.col	100	166	3.35	3	100	5	0.06	5	0	0.03	5	0	0.02
mug88_1.col	88	146	3.81	3	88	5	0.05	5	0	0.02	5	0	0.01
mug88_25.col	88	146	3.81	3	88	5	0.03	5	0	0.02	5	0	0.01
mulsol.i.1.col	197	3,925	20.33	49	65	51	0.03	51	0	0.01	51	0	0.01
mulsol.i.2.col	188	3,885	22.10	31	124	32	0.16	32	0	0.10	32	0	0.06
mulsol.i.3.col	184	3,916	23.26	31	125	32	0.14	32	0	0.11	32	0	0.06
mulsol.i.4.col	185	3,946	23.18	31	126	32	0.14	32	0	0.17	32	0	0.06
mulsol.i.5.col	186	3,973	23.09	31	127	32	0.16	32	0	0.13	32	0	0.06
myciel3.col	11	20	36.36	2	11	4	0.02	4	0	0.00	4	0	0.00
myciel4.col	23	71	28.06	2	23	5	0.01	5	0	0.00	5	0	0.00
myciel5.col	47	236	21.83	2	47	5	0.02	5	0	0.01	5	0	0.01
myciel6.col	95	755	16.91	2	95	5	0.42	5	0	0.25	5	0	0.07
myciel7.col	191	2,360	13.01	2	191	5	8.60	5	0	3.27	5	0	0.80
qg.order30.col	900	26,100	6.45	30	900	30	†	30	273	†	30	0	450.03
qg.order40.col	1,600	62,400	4.88	40	1,600	40	†	40	1,057	†	40	725	†
qg.order60.col	3,600	212,400	3.28	60	3,600	60	†	60	3,186	†	60	2,893	†
queen10_10.col	100	1,470	29.70	10	100	10	0.94	10	0	1.06	10	0	0.26

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
queen11_11.col	121	1,980	27.27	11	121	11	1.92	11	0	2.28	11	0	0.51
queen12_12.col	144	2,596	25.21	12	144	12	3.79	12	0	3.42	12	0	0.91
queen13_13.col	169	3,328	23.44	13	169	13	7.43	13	0	6.58	13	0	1.62
queen14_14.col	196	4,186	21.91	14	196	14	13.26	14	0	11.29	14	0	2.60
queen15_15.col	225	5,180	20.56	15	225	15	22.25	15	0	19.89	15	0	4.70
queen16_16.col	256	6,320	19.36	16	256	16	38.63	16	0	35.94	16	0	8.11
queen5_5.col	25	160	53.33	5	25	7	0.02	7	0	0.00	7	0	0.00
queen6_6.col	36	290	46.03	6	36	7	0.02	7	0	0.01	7	0	0.01
queen7_7.col	49	476	40.48	7	49	8	0.05	8	0	0.05	8	0	0.03
queen8_12.col	96	1,368	30.00	12	96	12	0.76	12	0	0.65	12	0	0.22
queen8_8.col	64	728	36.11	8	64	8	0.19	8	0	0.15	8	0	0.07
queen9_9.col	81	1,056	32.59	9	81	9	0.44	9	0	0.38	9	0	0.14
r1000.1.col	1,000	14,378	2.88	20	924	22	0.30	22	0	51.43	22	0	8.03
r1000.1c.col	1,000	485,090	97.12	91	1,000	33	†	35	947	†	35	947	†
r1000.5.col	1,000	238,267	47.70	234	985	74	†	76	707	†	84	679	†
r125.1.col	125	209	2.70	5	125	7	0.01	7	0	0.05	7	0	0.02
r125.1c.col	125	7,501	96.79	46	125	38	†	38	69	†	38	69	†
r125.5.col	125	3,838	49.52	36	122	38	10.44	38	0	6.19	38	0	4.45
r250.1.col	250	867	2.79	8	234	9	0.02	9	0	0.64	9	0	0.19
r250.1c.col	250	30,227	97.12	63	250	38	†	39	193	†	39	193	†
r250.5.col	250	14,849	47.71	65	245	67	493.65	67	0	145.97	67	0	88.03

Continued on next page

Graph name	$\|V\|$	$\|E\|$	$\rho(G)$	$\omega(G)$	V_{red}	opt_G	Time	opt_{seq}	i^*	Time	opt_{par}	i^*	Time
school1.col	385	19,095	25.83	14	363	18	†	18	208	†	18	203	†
school1_nsh.col	352	14,612	23.65	14	333	18	†	18	136	†	18	119	†
wap01a.col	2,368	110,871	3.96	41	2,288	40	†	40	1,383	†	41	1,133	†
wap02a.col	2,464	111,742	3.68	40	2,377	40	†	40	1,406	†	40	1,271	†
wap03a.col	4,730	286,722	2.56	40	4,717	40	†	40	3,678	†	40	3,448	†
wap04a.col	5,231	294,902	2.16	40	5,223	40	†	40	3,859	†	40	3,630	†
wap05a.col	905	43,081	10.53	50	693	50	215.35	50	0	210.41	50	0	59.32
wap06a.col	947	43,571	9.73	40	865	40	†	40	213	†	41	0	417.08
wap07a.col	1,809	103,368	6.32	40	1,724	42	†	42	828	†	42	600	†
wap08a.col	1,870	104,176	5.96	40	1,779	41	†	41	901	†	41	687	†
will199GPIA.col	701	6,772	2.76	6	701	9	1.39	9	0	109.85	9	0	20.58
zeroin.i.1.col	211	4,100	18.51	49	91	50	4.46	50	0	1.41	50	0	1.34
zeroin.i.2.col	211	3,541	15.98	30	137	32	0.95	32	0	0.55	32	0	0.47
zeroin.i.3.col	206	3,540	16.77	30	137	32	0.95	32	0	0.55	32	0	0.48

APPENDIX B

HADWIGER'S NUMBER COMPUTATIONAL RESULTS

B.1 Scale reduction

This table shows the sizes of two largest bicomponents and total number of bicomponents. Graph instances are from SNAP[1] and DIMACS X Challenge[2]. The columns are:

- *Graph* : graph file name;
- $|V|$: number of vertices;
- $|E|$: number of edges;
- $\tilde{\rho}(G)$: density times 100%;
- *# BCC* : number of biconnected components;
- $|C_1|$: size of the largest biconnected component;
- $|C_2|$: size of the second largest biconnected component;
- V_{red} : size of the biconnected 3-core obtained from the largest biconnected 2-core by eliminating vertices of degree 2 using the contraction of one of its incident edges.

Table B.1: Scale reduction computations.

Graph	$\|V\|$	$\|E\|$	$\tilde{\rho}(G)$	# BCC	$\|C_1\|$	$\|C_2\|$	V_{red}
SNAP							
amazon0302	262,111	899,792	0.00	7,849	248,019	107	241,418
amazon0312	400,727	2,349,869	0.00	20,241	378,225	58	362,338

Continued on next page

[1] https://snap.stanford.edu/data/
[2] https://www.cc.gatech.edu/dimacs10/

Graph	$\|V\|$	$\|E\|$	$\tilde{\rho}(G)$	# BCC	$\|C_1\|$	$\|C_2\|$	V_{red}
amazon0505	410,236	2,439,437	0.00	21,662	386,063	79	368,977
amazon0601	403,394	2,443,408	0.00	13,331	387,422	117	371,403
as20000102	6,474	12,572	0.06	2,458	4,009	4	1,525
as-caida20071105	26,475	53,381	0.02	10,195	16,264	4	5,587
as-skitter	1,696,415	11,095,298	0.00	238,721	1,443,769	214	1,186,973
ca-AstroPh	18,772	198,050	0.11	1,690	15,929	18	14,832
ca-CondMat	23,133	93,439	0.03	3,343	17,234	19	15,297
ca-GrQc	5,242	14,484	0.11	1,548	2,651	35	2,110
ca-HepPh	12,008	118,489	0.16	1,868	9,025	26	7,788
ca-HepTh	9,877	25,973	0.05	2,618	5,898	24	4,637
cit-HepPh	34,546	420,877	0.07	1,415	32,997	10	31,801
cit-HepTh	27,770	352,285	0.09	1,686	25,742	11	24,459
cit-Patents	3,774,768	16,518,947	0.00	678,955	3,090,253	12	2,690,920
com-amazon.ungraph	334,863	925,872	0.00	35,494	281,089	89	245,273
com-dblp.ungraph	317,080	1,049,866	0.00	68,360	211,409	33	174,924
com-lj.ungraph	3,997,962	34,681,189	0.00	840,916	3,120,286	95	2,703,852
com-orkut.ungraph	3,072,441	117,185,083	0.00	68,117	3,003,914	14	2,960,859
com-youtube.ungraph	1,134,890	2,987,624	0.00	673,661	452,060	17	288,538
email-Enron	36,692	183,831	0.03	12,093	20,416	70	17,775
email-EuAll	265,214	364,481	0.00	213,213	36,106	14	16,044
email-Eu-core	1,005	16,064	3.18	96	891	2	855
facebook_combined	4,039	88,234	1.08	90	3,698	191	3,623
loc-brightkite_edges	58,228	214,078	0.01	23,687	33,187	13	25,022
loc-gowalla_edges	196,591	950,327	0.00	56,260	137,519	15	108,410
p2p-Gnutella04	10,876	39,994	0.07	2,498	8,379	2	6,949
p2p-Gnutella05	8,846	31,839	0.08	2,012	6,830	4	5,769
p2p-Gnutella06	8,717	31,525	0.08	1,991	6,727	2	5,527
p2p-Gnutella08	6,301	20,777	0.10	1,766	4,535	2	3,668
p2p-Gnutella09	8,114	26,013	0.08	2,504	5,606	2	4,474
p2p-Gnutella24	26,518	65,369	0.02	10,990	15,519	2	11,566

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\tilde{\rho}(G)$	# BCC	$\|C_1\|$	$\|C_2\|$	V_{red}
p2p-Gnutella25	22,687	54,705	0.02	9,326	13,347	3	9,827
p2p-Gnutella30	36,682	88,328	0.01	16,479	20,192	3	14,95
p2p-Gnutella31	62,586	147,892	0.01	28,762	33,812	3	24,440
roadNet-CA	1,965,206	2,766,607	0.00	381,366	1,563,362	462	1,150,341
roadNet-PA	1,088,092	1,541,898	0.00	219,142	863,105	42	639,084
roadNet-TX	1,379,917	1,921,660	0.00	293,694	1,050,434	24,092	761,358
soc-Epinions1	75,879	405,740	0.01	39,120	36,111	34	25,913
soc-LiveJournal1	4,847,571	42,851,237	0.00	1,133,883	3,665,291	120	3,137,049
soc-sign-epinions	131,828	711,210	0.01	67,177	58,221	14	41,619
soc-Slashdot0811	77,360	469,180	0.02	29,867	47,217	9	36,232
soc-Slashdot0902	82,168	504,230	0.01	30,288	51,528	11	39,431
twitter_combined	81,306	1,342,296	0.04	5,258	75,400	175	70,809
web-BerkStan	685,230	6,649,470	0.00	62,295	489,296	13,728	448,822
web-Google	875,713	4,322,051	0.00	175,274	641,997	384	554,913
web-NotreDame	325,729	1,090,108	0.00	167,543	134,958	614	98,705
web-Stanford	281,903	1,992,636	0.01	29,470	181,906	4,907	165,024
wiki-Talk	2,394,385	4,659,565	0.00	1,769,165	622,315	8	274,486
wiki-topcats	1,791,489	25,444,207	0.00	6,387	1,784,918	39	1,741,596
wiki-Vote	7,115	100,762	0.40	2,307	4,786	2	4,154
DIMACS X							
adjnoun	112	425	6.84	11	102	2	90
af_shell10	1,508,065	25,582,130	0.00	1	1,508,065	0	1,508,065
af_shell9	504,855	8,542,010	0.01	1	504,855	0	504,855
as-22july06	22,963	48,436	0.02	8,010	14,939	4	5,051
astro-ph	16,706	121,251	0.09	1,905	12,672	18	11,668
audikw1	943,695	38,354,076	0.01	1	943,695	0	943,695
cage15	5,154,859	47,022,346	0.00	1	5,154,859	0	5,154,667
celegans_metabolic	453	2,025	1.98	16	423	6	412
citationCiteseer	268,495	1,156,647	0.00	44,467	220,937	40	187,936
coAuthorsCiteseer	227,320	814,134	0.00	48,393	142,245	42	121,361

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\tilde{\rho}(G)$	# BCC	$\|C_1\|$	$\|C_2\|$	V_{red}
coAuthorsDBLP	299,067	977,676	0.00	65,636	197,448	33	163,247
cond-mat-2003	31,163	120,029	0.02	4,685	21,772	22	19,266
cond-mat-2005	40,421	175,691	0.02	5,446	29,560	30	26,465
cond-mat	16726	47,594	0.03	3,285	9,842	18	8,502
coPapersCiteseer	434,102	16,036,720	0.02	11,577	401,268	93	397,669
coPapersDBLP	540,486	15,245,729	0.01	17,066	503,601	58	497,845
dolphins	62	159	8.41	10	53	2	46
email	1,133	5,451	0.85	157	976	3	855
football	115	613	9.35	1	115	0	115
G_n_pin_pout	100,000	501,198	0.01	54	99,942	2	99,735
hep-th	8,361	15,751	0.05	2,312	3,673	24	2,843
jazz	198	2,742	14.06	6	193	2	188
karate	34	78	13.90	3	28	6	18
ldoor	952,203	22,785,136	0.01	1	952,203	0	952,203
lesmis	77	262	8.95	18	54	7	47
netscience	1,589	2,742	0.22	465	134	29	108
PGPgiantcompo	10,680	24,316	0.04	5,992	3,670	28	2,573
polblogs	1,490	16,715	1.51	142	1,081	3	976
polbooks	105	441	8.08	1	105	0	103
power	4,941	6,594	0.05	1,688	3,040	50	1282
preferentialAttachment	100,000	499,985	0.01	1	100,000	0	100,000
rgg_n_2_15_s0	32,768	160,240	0.03	39	32,713	4	32,620
rgg_n_2_16_s0	65,536	342,127	0.02	53	65,446	15	65,318
rgg_n_2_17_s0	131,072	728,753	0.01	27	131,028	9	130,885
rgg_n_2_18_s0	262,144	1,547,283	0.00	54	262,067	8	261,897
rgg_n_2_19_s0	524,288	3,269,766	0.00	49	524,214	7	524,028
rgg_n_2_20_s0	1,048,576	6,891,620	0.00	56	1,048,498	10	1,048,281
rgg_n_2_21_s0	2,097,152	14,487,995	0.00	58	2,097,075	4	2,096,810
rgg_n_2_22_s0	4,194,304	30,359,198	0.00	67	4,194,223	4	4,193,955
rgg_n_2_23_s0	8,388,608	63,501,393	0.00	57	8,388,540	4	8,388,213

Continued on next page

Graph	$\|V\|$	$\|E\|$	$\tilde{\rho}(G)$	# BCC	$\|C_1\|$	$\|C_2\|$	V_{red}
rgg_n_2_24_s0	16,777,216	132,557,200	0.00	69	16,777,128	5	16,776,805
smallworld	100,000	499,998	0.01	1	100,000	0	100,000
thermal2	1,227,087	3,676,134	0.00	1	1,227,087	0	1,227,078
uk-2002	18,520,486	261,787,258	0.00	2,545,565	14,234,976	3,999	12,943,118

B.2 Upper and lower bounds

The next table demonstrates the lower bound obtained by Algorithm 12 and upper bound from Theorem 15 for coloring instances[3]. The columns are:

- *Graph file* : graph file name;
- $|V|$: number of vertices;
- $|E|$: number of edges;
- $\rho(G)$: density;
- $\chi(G)$: chromatic number;
- $\hat{h}(G)$: lower bound from Algorithm 12;
- *UB*: upper bound from Theorem 15.

Table B.2: Lower and upper bounds for Hadwiger's number on coloring instances.

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
1-FullIns_3	30	100	0.23	4	9	13
1-FullIns_4	93	593	0.14	5	21	33
1-FullIns_5	282	3,247	0.08	6	43	78

Continued on next page

[3] https://turing.cs.hbg.psu.edu/txn131/graphcoloring.html

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
1-Insertions_4	67	232	0.1	5	12	19
1-Insertions_5	202	1,227	0.06	6	25	46
1-Insertions_6	607	6,337	0.03	7	49	108
2-FullIns_3	52	201	0.15	5	13	18
2-FullIns_4	212	1,621	0.07	6	29	54
2-FullIns_5	852	12,201	0.03	7	69	152
2-Insertions_3	37	72	0.11	4	5	10
2-Insertions_4	149	541	0.05	5	15	29
2-Insertions_5	597	3,936	0.02	6	36	83
3-FullIns_3	80	346	0.11	6	16	24
3-FullIns_4	405	3,524	0.04	7	41	80
3-Insertions_3	56	110	0.07	4	5	12
3-Insertions_4	281	1,046	0.03	5	19	40
3-Insertions_5	1,406	9,695	0.01	6	51	130
4-FullIns_3	114	541	0.08	7	19	30
4-FullIns_4	690	6,650	0.03	8	53	110
4-Insertions_3	79	156	0.05	4	5	14
4-Insertions_4	475	1,795	0.02	5	23	52
5-FullIns_3	154	792	0.07	8	22	37
5-FullIns_4	1,085	11,395	0.02	9	67	145
abb313GPIA	1,557	53,356	0.04	8-10	96	323
anna	138	493	0.05	11	12	28
ash331GPIA	662	4,181	0.02	4	33	85
ash608GPIA	1,216	7,844	0.01	4	43	116
ash958GPIA	1,916	12,506	0.01	4	42	147
C2000.5	2,000	999,836	0.5	99-145	-	1,414
C2000.9	2,000	1,799,532	0.9	$\leq$400	-	1,897
C4000.5	4,000	4,000,268	0.5	107-259	-	2,828
david	87	406	0.11	11	13	26
DSJC1000.1	1,000	49,629	0.1	10-20	141	313

Continued on next page

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
DSJC1000.5	1,000	249,826	0.5	73-82	295	706
DSJC1000.9	1,000	449,449	0.9	216-222	515	948
DSJC125.1	125	736	0.09	5	23	36
DSJC125.5	125	3,891	0.5	17	52	88
DSJC125.9	125	6,961	0.9	44	77	118
DSJC250.1	250	3,218	0.1	4-8	45	78
DSJC250.5	250	15,668	0.5	26-28	96	177
DSJC250.9	250	27,897	0.9	72	143	236
DSJC500.1	500	12,458	0.1	9-12	79	156
DSJC500.5	500	62,624	0.5	43-47	166	353
DSJC500.9	500	112,437	0.9	123-126	265	474
DSJR500.1	500	3,555	0.03	12	19	79
DSJR500.1c	500	121,275	0.97	85	284	492
DSJR500.5	500	58,862	0.47	122	193	343
flat1000_50_0	1,000	245,000	0.49	15-50	293	700
flat1000_60_0	1,000	245,830	0.49	14-60	-	701
flat1000_76_0	1,000	246,708	0.49	14-81	-	702
flat300_28_0	300	21,695	0.48	28	104	208
fpsol2.i.1	496	11,654	0.09	65	67	150
fpsol2.i.2	451	8,691	0.09	30	32	129
fpsol2.i.3	425	8,688	0.1	30	32	130
games120	120	638	0.09	9	20	33
homer	561	1,628	0.01	13	24	46
huck	74	301	0.11	11	11	21
inithx.i.1	864	18,707	0.05	54	57	190
inithx.i.2	645	13,979	0.07	31	32	164
inithx.i.3	621	13,969	0.07	31	32	164
jean	80	254	0.08	10	10	20
latin_square_10	900	307,350	0.76	90-97	-	784
le450_15a	450	8,168	0.08	15	63	125

Continued on next page

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
le450_15b	450	8,169	0.08	15	64	125
le450_15c	450	16,680	0.17	15	87	181
le450_15d	450	16,750	0.17	15	87	182
le450_25a	450	8,260	0.08	25	63	126
le450_25b	450	8,263	0.08	25	64	126
le450_25c	450	17,343	0.17	25	91	185
le450_25d	450	17,425	0.17	25	89	185
le450_5a	450	5,714	0.06	5	54	104
le450_5b	450	5,734	0.06	5	54	104
le450_5c	450	9,803	0.1	5	68	138
le450_5d	450	9,757	0.1	5	69	137
miles1000	128	3,216	0.4	42	49	80
miles1500	128	5,198	0.64	73	78	102
miles250	128	387	0.05	8	10	22
miles500	128	1,170	0.14	20	23	47
miles750	128	2,113	0.26	31	34	64
mug100_1	100	166	0.03	4	4	13
mug100_25	100	166	0.03	4	4	13
mug88_1	88	146	0.04	4	4	12
mug88_25	88	146	0.04	4	4	12
mulsol.i.1	197	3,925	0.2	49	51	87
mulsol.i.2	188	3,885	0.22	31	33	87
mulsol.i.3	184	3,916	0.23	31	33	87
mulsol.i.4	185	3,946	0.23	31	33	88
mulsol.i.5	186	3,973	0.23	31	32	88
myciel3	11	20	0.36	4	5	6
myciel4	23	71	0.28	5	8	11
myciel5	47	236	0.22	6	14	21
myciel6	95	755	0.17	7	24	37
myciel7	191	2,360	0.13	8	38	67

Continued on next page

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
qg.order100	10,000	990,000	0.02	100	-	1,401
qg.order30	900	26,100	0.06	30	59	226
qg.order40	1,600	62,400	0.05	40	-	350
qg.order60	3,600	212,400	0.03	60	-	647
queen10_10	100	2,940	0.59	11	29	76
queen11_11	121	3,960	0.55	11	34	89
queen12_12	144	5,192	0.5	12	37	101
queen13_13	169	6,656	0.47	13	43	115
queen14_14	196	4,186	0.22	14	47	90
queen15_15	225	5,180	0.21	15	52	101
queen5_5	25	160	0.53	5	13	18
queen6_6	36	290	0.46	7	16	24
queen7_7	49	476	0.4	7	20	30
queen8_12	96	1,368	0.3	12	28	51
queen8_8	64	728	0.36	9	23	37
queen9_9	81	1,056	0.33	10	26	45
r1000.1	1,000	14,378	0.03	20	45	165
r1000.1c	1,000	485,090	0.97	96-98	534	985
r1000.5	1,000	238,267	0.48	234	-	690
r125.1	125	209	0.03	5	6	9
r125.1c	125	7,501	0.97	46	84	122
r125.5	125	3,838	0.5	36	55	87
r250.1	250	867	0.03	8	10	36
r250.1c	250	30,227	0.97	64	153	246
r250.5	250	14,849	0.48	65	103	172
school1	385	19,095	0.26	14	97	194
school1_nsh	352	14,612	0.24	14	86	170
wap01a	2368	110,871	0.04	$\geq$41	148	467
wap02a	2464	111,742	0.04	$\geq$40	-	469
wap03a	4730	286,722	0.03	$\geq$40	-	752

Continued on next page

Graph file	$\|V\|$	$\|E\|$	$\rho(G)$	$\chi(G)$	$\hat{h}(G)$	UB
wap04a	5231	294,902	0.02	40-42	-	762
wap05a	905	43,081	0.11	50	111	291
wap06a	947	43,571	0.1	$\geq$40	110	293
wap07a	1809	103,368	0.06	40-41	-	452
wap08a	1870	104,176	0.06	$\geq$40	-	453
will199GPIA	701	6,772	0.03	7	42	111
zeroin.i.1	211	4,100	0.19	49	51	89
zeroin.i.2	211	3,541	0.16	30	33	83
zeroin.i.3	206	3,540	0.17	30	33	83

B.3 Enumerating partitions

Algorithm to list partitions in lexicographic order to allow efficient pruning of infeasible ones[4].

Algorithm 13 Enumerate partitions in lexicographical order

```
1: LB = 0
2: procedure PARTITIONS(U, left, right, i, parts)
3:     if U = ∅ then
4:         if |parts| > LB then
5:             LB = |parts|                                  ▷ better solution found
6:     for j ∈ [i; |U|] do
7:         left = left ∪U[j]
8:         if left violates contiguity and proximity then
9:             continue for loop
10:        parts = parts ∪ left
11:        U' ← U from j + 1
12:        PARTITIONS(U', ∅, ∅, 0, parts)
13:        remove last element from parts
14:        PARTITION(U, left, right, j + 1, parts)
15:        if i = 0 then
16:            return
17:        remove last element from left
18:        right = right ∪U[j]
19:    remove last |U| − i elements from right
20: return LB as h(G)
21: U = {1, . . . , |V|}
22: PARTITIONS(U, ∅, ∅, 0, ∅)
```

[4]We thank Mykyta Makovenko for his improvements and comments about the algorithm.